НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Автоматика і Управління в Технічних Системах

«На правах рукопису»
УДК _____

До захисту допущено:
Завідувач кафедри
_____ Олександр Ролік
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-науковою програмою «Інженерія програмного забезпечення комп'ютерних систем»

зі спеціальності 121 «Інженерія програмного забезпечення»

на тему: «Веб-застосунок для онлайн управління та взаємодії з працівниками»

Виконав:
студент VI курсу, групи ІТ-91мн
Черненко Максим _____

Науковий керівник:
д.т.н., професор кафедри АУТС
Корнієнко Богдан

Рецензент:
Доцент кафедри технічних та програмних засобів автоматизації ІХФ, к.т.н., доцент
Ладієва Леся
_____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2021 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Автоматика і Управління в Технічних Системах

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма - «Інженерія програмного забезпечення комп'ютерних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр Ролік
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Черненко Максиму Юрійовичу

1. Тема дисертації «Веб-застосунок для онлайн управління та взаємодії з працівниками», науковий керівник дисертації Корніенко, д.т.н. професор кафедри АУТС, затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом дисертації 11.05.2021

3. Об'єкт дослідження: *crm-системи*

4. Предмет дослідження: *процес управління взаємовідносинами з співробітниками*

5. Перелік завдань, які потрібно розробити: *знайти та проаналізувати існуючі рішення та обрати алгоритми для подальшої реалізації; Розробити програмне забезпечення, його опис та інструкцію до нього*

6. Орієнтовний перелік графічного (ілюстративного) матеріалу

7. Орієнтовний перелік публікацій

8. Консультанти розділів дисертації

Розділ		Підпис, дата

	Прізвище, ініціали та посада консультанта	завдання видав	завдання прийняв

9. Дата видачі завдання 01.02.2021_____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1		1.02.2021	
2			
3			
4			
		11.05.2021	

Студент Черненко М. Ю.

Науковий керівник Корніенко Б. Я.

РЕФЕРАТ

Актуальність теми: На сьогодняшній день IT компанії використовуюсь декілька сервісів одночасно - один для ставлення завдань співробітникам, інший для листування, третій для отримання фідбеку про колег , для таймтрекінгу і т.д. Моя ідея полягає в тому, щоб поєднати усі ці процеси в один зручний сервіс. Так як зараз багато компаній у зв'язку з карантином переходять на full-remote і бачать в цьому майбутнє, ця тема, на мою думку, надзвичайно актуальна. Повний контроль і взаємодія з працівниками в одному сервісі.

Мета дослідження: основна мета полягає створенні програмного забезпечення для управління співробітниками та взаємодії між ними.

Об`єкт дослідження: управління та взаємодія, crm-системи, системи управління проєктами, системи запису робочого часу.

Для реалізації поставленої мети сформульовано наступні **завдання:**

– дослідження та аналіз існуючих рішень та вибір алгоритмів для подальшої реалізації;

– розробка архітектури програмного забезпечення, бібліотек, необхідних для його роботи;

– розробка програмного забезпечення, його опису та інструкції до нього.

Предмет дослідження: вебзастосунок для управління взаємовідносинами з кліентами.

Наукова новизна: наукова новизна роботи полягає у розробці веб-застосунку, спрямованого на формування, розвиток та підвищення ефективності співробітників, орієнтованих на вдосконалення взаємодії та результат.

ABSTRACT

Actuallity: Today, IT companies use several services simultaneously - one to assign tasks to employees, another for correspondence, the third to get feedback about colleagues, for time tracking, etc. My idea is to combine all these processes into one convenient service. Since many companies are now moving to full-remote quarantine and see a future in this, I think this topic is extremely relevant. Full control and interaction with employees in one service.

The aim of research: the main goal is to create software for employee management and interaction between them.

The object of research: management and interaction, crm-systems, project management systems, time and attendance systems.

To achieve this goal, the following tasks are formulated:

– search and analysis of existing solutions and selection of algorithms for further implementation;

– develop and substantiate mathematical software for analysis;

– development of software architecture, libraries necessary for its work;

– develop software, its description and instructions for it.

Research subject: web application for customer relationship management.

Research originality: The scientific novelty of the work is the development of a web application aimed at the formation, development and efficiency of employees focused on improving interaction and results.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Bcrypt – криптографічна хеш-функція формування ключа.

Exsblowfish – алгоритм хешування паролів.

MongoDB – крос-платформна, документо-орієнтована база даних.

NoSQL – нереляційна база даних.

React – JavaScript-бібліотека з відкритим вихідним кодом для розробки призначених для користувача інтерфейсів.

MVC – це патерн проектування веб-додатків, який включає в себе три окремих компонента. Три окремих компонента розділені на модель даних програми, призначений для користувача інтерфейс і логіку взаємодії користувача з системою.

JSX – розширення мови JavaScript.

Redux – бібліотека для JavaScript з відкритим вихідним кодом, призначена для управління станом додатки.

JavaScript – мова програмування, який додає інтерактивність на ваш веб-сайт.

GUI – графічний користувальницький інтерфейс.

Webpack – складальник модулів JavaScript з відкритим вихідним кодом.

REST – архітектурний стиль взаємодії компонентів розподіленого додатка в мережі.

API – сукупність різних інструментів, функцій, реалізованих у вигляді інтерфейсу для створення нових додатків, завдяки якому одна програма буде взаємодіяти з іншого.

JSON – текстовий формат обміну даними, заснований на JavaScript.

JWT – це відкритий стандарт для створення токенов доступу, заснований на форматі JSON.

ВСТУП

Актуальність теми: Будь-який бізнес пов'язаний з об'єднанням ресурсів для створення певного продукту. Безпосереднє перетворення ресурсів завжди передбачає участь працівників як особливого виду ресурсів. Тому для будь-якої організації персонал складає основу бізнесу. Крім того, працівники можуть виступати учасниками управління, якщо мова йде про управлінні людськими ресурсами.

Необхідність ефективного управління персоналом визначається значимістю людських ресурсів для досягнення ефективності в управлінні бізнесом в цілому. Людські ресурси виступають основою ведення бізнесу, але рішення щодо способу їх найкращого використання, незалежно від кадрового потенціалу приймаються керівництвом організації. Оскільки людські ресурси як основа управління виступають одним з ключових елементів пристосування організації до зовнішнього середовища, одночасно працівники можуть розумітися як один з факторів зовнішнього середовища, а не тільки внутрішнього середовища компанії, управління людськими ресурсами відноситься до стратегічного рівня управління бізнесом в цілому. При цьому функціональна стратегія по персоналу організації відноситься до найбільш значимих елементів стратегічного цілепокладання в організаціях.

Вироблення функціональної стратегії щодо персоналу організації передбачає вирішення питань щодо місця працівників в управлінні організацією, тобто стратегія управління персоналом може ставитися тільки до працівників або до формування людських ресурсів як елемента управління компанією. Крім того, стратегічне управління в відношенні персоналу, навіть якщо воно не опосередковано залученням пересічних співробітників в управління бізнесом, в будь-якому випадку є однією з специфічних систем в рамках загальної системи управління. Стратегія управління персоналом повинна охоплювати найрізноманітніші аспекти, при наявності таких підсистем як управління наймом,

винагороди та мотивація, оцінка та розвиток персоналу, конкретний склад підсистем управління персоналом формується на стратегічному рівні.

На сьогодняшній день IT компанії використовуюсь декілька сервісів одночасно для управління та взаємодії з працівниками - один для ставлення завдань співробітникам, інший для листування, третій для отримання фідбеку про колег , для таймтрекінгу і т.д. Моя ідея полягає в тому, щоб поєднати усі ці процеси в один зручний сервіс. Так як зараз багато компаній у зв'язку з карантином переходять на full-remote і бачать в цьому майбутнє, ця тема, на мою думку, надзвичайно актуальна. Повний контроль і взаємодія з працівниками в одному сервісі.

Мета дослідження: основна мета полягає створенні програмного забезпечення для управління співробітниками та взаємодії між ними.

Об`єкт дослідження: управління та взаємодія, crm-системи, системи управління проєктами, системи запису робочого часу.

Для реалізації поставленої мети сформульовано наступні **завдання:**

– пошук та аналіз існуючих рішень та вибір алгоритмів для подальшої реалізації;

– розробка та обгрунтування математичного забезпечення для аналізу;

– розробка архітектури програмного забезпечення, бібліотек, необхідних для його роботи;

– розробка програмного забезпечення, його опису та інструкції до нього.

Предмет дослідження: вебзастосунок для управління взаємовідносинами з кліентами.

Наукова новизна: наукова новизна роботи полягає у розробці веб-застосунку, спрямованого на формування, розвиток та підвищення ефективності співробітників, орієнтованих на вдосконалення взаємодії та результат.

1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Загальні положення

Основної ціллю є написання веб застосунку для онлайн управління та взаємодії з працівниками.

Мета розробки - узагальнення теоретичних знань отриманих впродовж учбового процессу для вирішення управління та взаємодії з працівниками з використанням API зовнішнього сервісу.

Призначення розробки – поліпшення зручності управління проєктами та взаємодії працівників.

Задачі:

- менеджування співробіниками;
- таймтрекінг роботи співробітників;
- чат для співробітників;
- отримання фідбеку про співробітників;
- ведення історії виконаних завдань співробітниками.

Цілі:

- надати користувачеві інтерфейс для поставлених задач;
- зробити можливим зручний аналіз статистики;
- та управління працівниками для топ-менеджменту;
- надати користувачеві зручний інтерфейс партнерів.

Програмний продукт, що задовольняє вимогам повинен бути написаний на мові, що дозволяє виконувати програму на серверному середовищі. Він повинен підтримувати HTTP запити по протоколу REST, з кодуванням данних за допомогою JSON. За необхідності роботи з зовнішніми сервісами, коммунікація повинна відбуватися по HTTP.

Для забезпечення високої доступності та зручності масштабування програмний продукт можна розділити на окремі сервіси, що відповідають за окремі процеси.

1.2 Змістовний опис і аналіз предметної області

Веб-застосунок, який поєднує в собі менеджування компанією та взаємодію співробітників між собою. На сьогодняшній день IT компанії використовуюсь декілька сервісів одночасно - один для ставлення завдань співробітникам, інший для листування, третій для отримання фідбеку про колег , для таймтрекінгу і т.д. Моя ідея полягає в тому, щоб поєднати усі ці процеси в один зручний сервіс.

Так як зараз багато компаній у зв’язку з карантином переходять на full-remote і бачать в цьому майбутнє, ця тема, на мою думку, надзвичайно актуальна. Повний контроль і взаємодія з працівниками в одному сервісі.

1.3 Аналіз успішних IT-проектів

Серед систем менеджування та взаємодії з співробітниками досить популярними є такі системи, як – Trello(менеджування), Slack(взаємодія з співробітниками), Microsoft Teams.

Trello — бесплатна багатоплатформна система управління проектами, розроблена Fog Creek Software. Вона використовує парадигму керування проектами, відому як канбан. Проекти зображуються дошками, що містять списки. Списки містять картки, якими зображуються задачі. Картки повинні переходити з попереднього списку до наступного (за допомогою перетягування), таким чином зображаючи рух якоїсь функції від ідеї, аж до тестування. Картці може бути присвоєно відповідальних за неї користувачів. Користувачі та дошки можуть об'єднуватись в команди[1].

Slack — це додаток для групового спілкування, що дозволяє обмінюватися повідомленнями в режимі реального часу в приватних групах, а також особистих і групових чатах, організованих за темами. Головна перевага Slack — можливість зручно вести робоче листування, залишаючи інші месенджери, такі як Whatsapp, Facebook Messenger, Telegram чи Viber, для особистого життя. При цьому, якщо в робочому чаті в тих же Telegram або Viber з'являться одночасно кілька питань, то їх обговорення перемішається в одній стрічці, в той час як в Slack є зручна функція тредів (впорядкованих гілок листування), що дає можливість структурувати повідомлення[2].

Microsoft Teams — центр для командної роботи в Office 365 від Microsoft, який інтегрує користувачів, вміст і засоби, необхідні команді для ефективнішої роботи. Застосунок об'єднує все в спільному робочому середовищі, яке містить чат для нарад, файлообмінник та корпоративні програми.[3]

1.4 Розроблення функціональних вимог

Таблиця 1.1 – Функціональні вимоги до системи

Варіант використання	Опис	Пріорітет
Додання задач через веб інтерфейс	1.Система повинна мати можливість додавати нові задачі.	Високий
Перегляд списку співробітників через веб-інтерфейс	2. Система повинна надавати користувачу список працівників	Високий

Перегляд історії зроблених задач у веб-інтерфейсі	3. Система повинна надавати користувачу результат виконаних задач.	Високий
Листування з працівниками через веб-інтерфейс	4. Система повинна надавати користувачу можливість листування через веб-інтерфейс.	Високий

1.5 Постановка завдання

Мета розробки - узагальнення теоретичних знань отриманих впродовж учбового процессу для вирішення проблеми управління та взаємодії з працівниками з використанням API зовнішнього сервісу.

Призначення розробки – поліпшення зручності управління проєктами та взаємодії працівників.

Задачі:

- менеджування співробіниками;
- таймтрекінг роботи співробітників;
- чат для співробітників;
- отримання фідбеку про співробітників;
- ведення історії виконаних завдань співробітниками.

1.6 Висновки по розділу

У цьому розділі було описано та проаналізовано предметну область розробки. Було виділено успішні IT- проекти у даній області та виконано

порівняння даного комплексу задач с готовими продуктами, де було розглянуто їх переваги та недоліки, виконано порівняння з даним комплексом задач та надано у вигляді таблиці.

2 ТЕОРИТИЧНІ ОСНОВИ УПРАВЛІННЯ ПЕРСОНАЛОМ В КОМПАНІЇ

2.1 Управління персоналом в сучасних умовах: поняття, суть, мети, завдання, функції.

Кожен з варіантів розуміння системи управління персоналом у взаємозв'язку з менеджментом є обгрунтованим, оскільки в рамках концепції управління персоналом як розуміння системи управління працівниками, що не залежить від інших елементів управління організацією, більш повно розкриваються окремі елементи системи управління персоналом.

Якщо говорити про систему управління персоналом як однієї з підсистем управління, можна більш обгрунтовано встановити взаємозв'язок між персоналом і іншими складовими діяльності організації. При цьому слід враховувати, що в рамках концепції управління людськими ресурсами персонал розуміється не тільки як елемент поточного управління діяльністю, а й як елемент стратегічного управління організацією. Тому і система управління людськими ресурсами повинна розумітися як один з елементів стратегічного управління організацією [4].

Для наочності співвідношення понять системи управління персоналом і системи управління людськими ресурсами як елементів управління організацією може бути представлено у вигляді рис. 2.1.

Рисунок 2.1 - Співвідношення понять системи управління персоналом і системи управління людськими ресурсами як елементів управління організацією

З рис. 2.1 видно, що система управління персоналом організації забезпечує участь працівників у виробничій діяльності як основну мету використання персоналу, крім того, працівники беруть участь в фінансовому управлінні, відповідають за маркетингову діяльність.

З точки зору управління людськими ресурсами додатково слід враховувати створення ініціатив для розвитку бізнесу, які сприяють більш ефективному здійсненню виробничої діяльності, створенню конкурентних переваг, що сприяє збільшення вартості компанії. Як наслідок, система управління персоналом як частина менеджменту повинна, як мінімум, забезпечувати участь працівників у

виробничій діяльності, сприяти виконанню обов'язків в області маркетингу і управління фінансовою діяльністю організації [5]. У науковій літературі представлені різні підходи до визначення поняття системи управління персоналом, наведені в табл. 2.1.

Таблиця 2.1 - Підходи до визначення поняття системи управління персоналом організації

Автор	Визначення	Переваги	Недоліки
А.Я. Кібанов	"Система управління персоналом передбачає формування цілей, функцій, організаційної структури управління персоналом, вертикальних і горизонтальних функціональних взаємозв'язків керівників і фахівців в процесі обгрунтування, вироблення, прийняття і реалізації управлінських рішень »[14]	- відзначається мета системи управління персоналом - система управління персоналом взаємопов'язана з іншими системами управління організацією - вказується на взаємозв'язок з виробленням і реалізацією рішень	- не вказується на механізми, за рахунок яких персонал бере участь в управлінні організацією - вказується тільки на спільні цілі системи управління персоналом, але не враховується співвідношення поточних цілей управління працівниками і цілей на перспективу
М. Армстронг	«Сукупність прийомів, методів, технологій організації роботи з персоналом »[6]	- вказується на методи і прийоми - система управління персоналом повинна бути організована	не визначені цілі системи управління персоналом - Відсутнє вказівка на взаємозв'язок між

			системою управління персоналом і керуванням організацією в цілому
Х.В. Сударкіна	«Сукупність взаємопов'язаних елементів організаційної, економічної, соціально-психологічної природи » [7]	- відзначається наявність безлічі елементів, складових систему управління персоналом - виділяється безліч факторів, визначальних формування системи управління персоналом	- Відсутнє вказівка на конкретні цілі системи управління персоналом - не враховується взаємозв'язок з керуванням організацією - недостатньо детально виражена роль працівників у системі управління персоналом
С.В. Вітик	«Сукупність методів роботи з працівниками в організації »[8]	управління персоналом грунтується на використанні визначених методів	- не вказується на взаємозв'язок елементів, що становлять систему управління персоналом - Відсутнє зазначення заходів, якими досягаються цілі системи управління

			персоналом у взаємозв'язку з керуванням організацією

З табл. 2.1 видно, що в науковій літературі система управління персоналом може розумітися як з точки зору формування взаємозв'язку з менеджментом організації, так і з позицій розуміння без урахування взаємозв'язку з керуванням. Більш обгрунтованою видається позиція, відповідно до якої система управління персоналом являє собою частину системи управління організацією в цілому.

Цілі системи управління персоналом пов'язані з загальними цілями

організації. Оскільки організації необхідно залучення працівників в досягнення загальних цілей, формуються окремі цілі, пов'язані з управлінням персоналом. Ці цілі пов'язані із заохоченням працівників до більш ефективної діяльності та розвитком кадрового потенціалу. Крім того, мети системи управління персоналом повинні включати в себе організаційні аспекти, пов'язані, в тому числі, з регулюванням окремих складових системи управління персоналом. Цілі системи управління персоналом організації по відношенню до працівникові представлені на рис. 2.2.

З точки зору працівника компанії необхідне створення таких умов, в яких формується задоволеність працею у взаємозв'язку з виконанням працівником трудових обов'язків як умови досягнення цілей організації. Тому, в першу чергу, організації необхідно запропонувати працівникові таке винагороду, яке заохочуватиме його до більш активної участі в досягненні цілей організації.

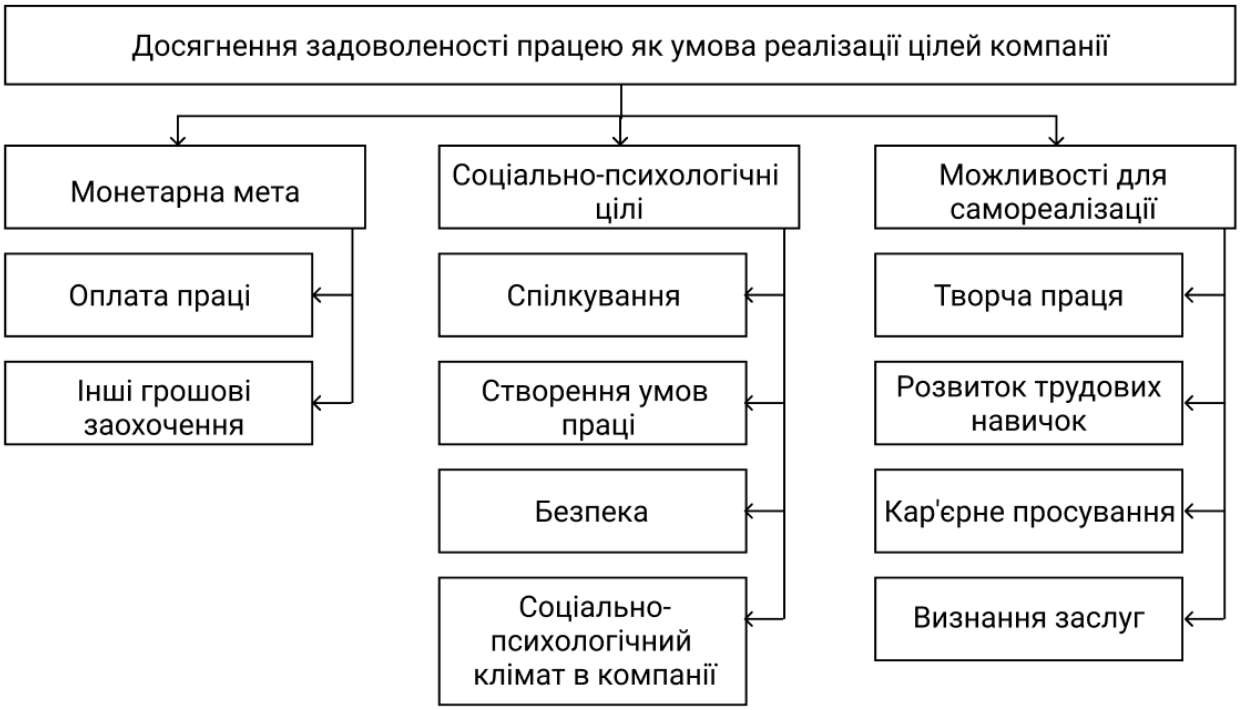

Рисунок 2.2 - Цілі системи управління персоналом по відношенню до працівника

Це винагорода повинна бути диференційованою в залежності від характеру участі в досягненні цілей. Крім того, задоволеність працівника працею знаходиться під взаємозв'язку з соціально-психологічними цілями. Працівник не може діяти ефективно в умовах несприятливого соціально-психологічного клімату. Крім того, йому необхідно запропонувати безпечні умови праці, забезпечити можливість для спілкування з іншими працівниками [28].

Оскільки працівник орієнтується не тільки на отримання грошових винагород, а й на самореалізацію, система управління персоналом повинна передбачати можливості для кар'єрного росту працівника, нарівні з визнанням заслуг і розвитком трудових навичок як основи для управління професійним просуванням працівника [15]. Цілі системи управління персоналом з точки зору організації представлені на рис. 2.3.

Організація орієнтується на отримання прибутку, тому система управління персоналом повинна забезпечувати можливості використання працівників для

досягнення цієї мети, а також створювати умови для підвищення ефективності використання трудового потенціалу на перспективу.[10]

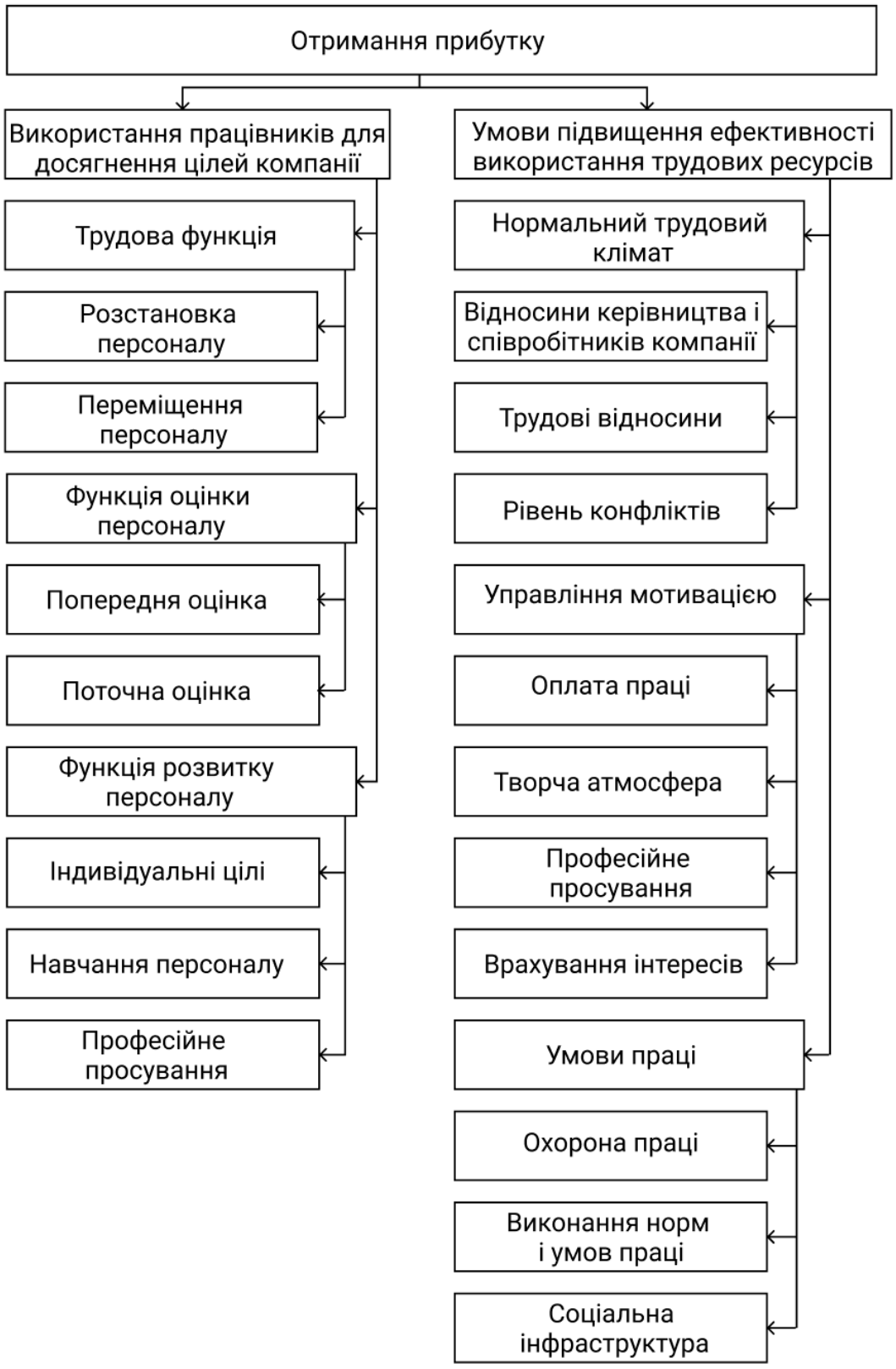

Рисунок 2.3 - Цілі системи управління персоналом з точки зору Організації

У плані використання персоналу найбільше значення має трудова функція, пов'язана з найкращим використанням співробітника виходячи з його навичок і потенціалу. Оскільки організації необхідно розташовувати об'єктивними відомостями про навички і потенціал працівників, система управління персоналом повинна забезпечувати оцінку працівників. З урахуванням результатів оцінки визначаються перспективи кращого використання працівників, тому система управління персоналом повинна забезпечувати його розвиток, включаючи встановлення індивідуальних цілей по взаємозв'язку з навчанням і просуванням по кар'єрних сходах. Для підвищення ефективності використання трудових ресурсів система управління персоналом повинна сприяти створенню сприятливого соціально-психологічного клімату в колективі.

Дивлячись на те, що досягнення цілей організації залежить від взаємин між співробітниками і керівництвом, необхідно формування взаємодії між цими суб'єктами, задіяними в досягненні цілей організації. Повинні, перш за все, формуватися нормальні трудові відносини.

Крім того, система управління персоналом повинна забезпечувати раціональне управління мотивацією праці, найбільш тісно пов'язане з цілями системи управління персоналом по відношенню до працівників. На додаток до винагород, організації необхідно управляти створенням творчої атмосфери, враховувати інтереси працівників і забезпечувати просування [39].

Управління персоналом в будь-якій організації базується на принципах корпоративної культури, професійних навичках персоналу тощо. Реалізувати потенціал працівників, поліпшити професійні якості персоналу - в цьому полягає основна діяльність служби управління. Як свідчить практика, служба управління персоналом вирішує такі організаційні завдання: оптимізація внутрішніх відносин в колективі, узгодження поставлених завдань і загальної стратегії розвитку організації, створення умов для соціальної захищеності персоналу,

підготовка і перепідготовка професійних кадрів, підвищення ефективності управління персоналом, виявлення та припинення конфліктів.

У вітчизняній науковій літературі представлені найрізноманітніші принципи управління персоналом. Однак більшість авторів сходяться на наступних. Принципи побудови системи управління персоналом діляться на дві великі групи: принципи, що мають безпосереднє відношення до формування служби, і загальні принципи функціонування служби. Розглянемо докладніше принципи управління персоналом в організації. Якщо служба управління персоналом ефективна, тоді витрати на її діяльність не будуть занадто високі і буде реалізовуватися принцип економічності. Якщо йде мова про принцип первинності функції управління персоналом, значить, має місце певна система і підсистеми в управлінні персоналом. Оптимальне співвідношення чисельності персоналу організації і технологічних можливостей виробництва реалізу принцип оптимізації.

Принцип ієрархічної підпорядкованості має на увазі наявність декількох рівнів підпорядкування.

Принцип прогресивності зобов'язує менеджера з персоналу активно застосовувати в своїй роботі прогресивні принципи і методи управління персоналом.

Принцип автономності наділяє цю службу повноваженнями в прийнятті рішень в рамках своєї безпосередньої діяльності, не вдаючись кожен раз до вищестоящого начальства.

Принцип перспективності дозволяє керівництву відділу по управлінню персоналом планувати свою діяльність з урахуванням перспективи розвитку організації в цілому.

Принцип оперативності - рішення повинні прийматися в строго встановлені терміни. Є також принципи адаптивності, концентрації, точності, спеціалізації, узгодженості, наступності та ін. [10]

Процес управління персоналом, включає в себе ряд важливих функцій, необхідних для ефективної роботи підприємства. Управління персоналом має сприяти підвищенню адаптаційних здібностей підприємства в умовах мінливого зовнішнього середовища.

В рамках управління умовами праці система управління персоналом повинна сприяти не тільки охорону праці, але формувати норми і умови праці, а також сприяти розвитку соціальної інфраструктури. Можуть бути виділені наступні функції управління персоналом [14]:

- планування персоналу, під яким розуміється визначення потреби організації в певній кількості працівників, характеризуються необхідної організації кваліфікацією і професійними знаннями. Планування персоналу проводиться з урахуванням часу його використання. Планування кадрів, як правило, розглядається в якості вихідної функції кадрового менеджменту, оскільки за часом вона випереджає всі інші функції управління персоналом;

- визначення способів найму працівників. Реалізація даної функції передбачає визначення джерел залучення працівників, способів їх залучення в організацію, а також розподіл залучених працівників між внутрішніми і зовнішніми джерелами. При створенні нової організації відповіді на дані питання даються ще на стадії проектування, оскільки залучення працівників, необхідних організації, не завжди можливо внаслідок відмінностей в ринках праці окремих регіонів і територій, що визначає необхідність планування способів залучення працівників ще на етапі проектування бізнесу. Також визначення способів найму працівників повинно враховувати і вартість робочої сили, оскільки вона може суттєво відрізнятися в залежності від джерел;

- маркетинг персоналу. Завданням маркетингу персоналу є забезпечення попиту на робочі місця в даній організації з боку працівників, які найбільшою мірою відповідають її потребам в трудових ресурсах. Перш за все, маркетинг персоналу спрямований на залучення висококваліфікованих фахівців і керівників. Маркетинг персоналу буває як зовнішнім, орієнтованим на залучення

працівників із зовнішніх джерел, так і внутрішнім, який передбачає спонукання працівників організації до підвищення свого професійного рівню для просування по кар'єрних сходах [27];

- підбір, відбір і розстановка персоналу. Передбачає залучення в організацію працівників, в найбільшою мірою відповідає вимогам до трудових ресурсів. Передбачає залучення потенційних працівників, вибір з їх числа тих, хто найбільшою мірою відповідає запитам організації до робочої сили, і їх розстановку. значимість даної функції визначається зростаючою ціною на трудові ресурси з одночасним підвищенням вимог до працівника;

- адаптація і професійний розвиток працівників організації. Передбачає забезпечення входження працівника в колектив організації після найму на роботу з метою максимально швидкого включення в професійну діяльність. Навчання працівників передбачає цілеспрямоване отримання ними нових професійних знань з метою найкращого виконання обов'язків та максимізації трудового потенціалу організації [7];

- планування кар'єри. Ця функція важлива як для окремих працівників організації, оскільки відповідає їхнім потребам, так і для компанії, оскільки вона забезпечує максимізацію використання трудового потенціалу працівника;

- мотивація і стимулювання праці працівників. Передбачає виявлення мотивів співробітників до сумлінної та ініціативному праці, а також створення системи заходів заохочення, які забезпечують найкращу віддачу від кожного працівника при прийнятному рівні витрат організації. В основі ефективного стимулювання праці персоналу лежить збіг організаційних і особистих цілей, яке забезпечується за рахунок всього комплексу можливих заходів заохочення, як матеріальних, так і нематеріальних;

- керівництво персоналом. Керівництво персоналом забезпечує об'єднання зусиль усіх працівників для досягнення поставлених перед організацією цілей. Основу керівництва працівниками становить їх мотивація, однак, також керівництво персоналом забезпечує об'єднання працівників, координацію їх

зусиль, а також взаємозв'язок і інтеграцію. В результаті досягається створення єдиної системи управління працівниками компанії [25];

- управління витратами на персонал. Передбачає визначення витрат на управління персоналом організації за окремими напрямами управління персоналом. Як правило, виділяються витрати на оплату праці працівників, обов'язкові відрахування в державні позабюджетні фонди, навчання співробітників організації і т.д .;

- організація робочого місця. Ця функція управління персоналом організації передбачає створення умов для максимального використання трудового потенціалу працівника. Ефективна організація робочого місця передбачає виявлення всіх факторів, які визначають ефективність використання трудового потенціалу конкретного працівника, а також можливі умови для підвищення віддачі від використання праці даного співробітника організації [22];

- забезпечення оптимального режиму праці. В цю функцію входить складання графіків роботи і відпусток, заміна в разі хвороби або прогулів і т.п. Для належного виконання даної функції необхідна координація різних напрямків управління персоналом, проте, її реалізація забезпечує оптимальний трудовий ритм організації, створює умови для економії значних коштів організації, а також підвищує задоволеність трудовою діяльністю в організації;

- вивільнення персоналу. Крім оформлення кадрових документів, пов'язаних з вибуттям працівника, дана функція передбачає і ряд інших напрямків. Зокрема, вона включає в себе різні форми адаптації чисельності працівників і їх використання для потреб організації. Це може бути перегляд понаднормових робіт, переклад працівників на неповний робочий день або неповний робочий тиждень, тимчасове звільнення і т.д .;

- кадрове діловодство. Ця функція управління персоналом передбачає збір, зберігання і облік кадрових документів. Також систематизується інформація про анкетні дані працівника, його стаж, підвищенні кваліфікації і т.д. Необхідність даної функції визначається потребою в ефективному використанні персоналу

організації. Сьогодні її ефективність значно підвищується за рахунок використання комп'ютерних систем і сучасних технологій [18];

- управління інформацією. Діяльність великих організацій передбачає наявність різної інформації, яка вимагає систематизації та передачі кінцевим одержувачам. З точки зору управління персоналом управління інформацією передбачає її доведення до конкретних працівників у необхідному обсязі і в необхідний для цього час. Також важливою є координація розподілу інформації між працівниками;

- оцінка ефективності використання трудового потенціалу організації. Ця функція передбачає проведення атестації працівників, а також їх ділову оцінку [15];

- забезпечення трудової дисципліни і підтримання порядку в організації. Ця функція передбачає як вплив на співробітників зі боку керівників, так і розвиток у працівників самоконтролю;

- управління конфліктами. Передбачає формування в організації клімату, який виключає виникнення деструктивних конфліктних ситуацій, а також забезпечує конструктивне, по можливості безболісне вирішення конфліктів;

- регулювання трудових відносин. Управління персоналом базується на положеннях трудового законодавства і організаційних норм, які закріплюються у внутрішніх документах організації;

- налагодження партнерських відносин з різними громадськими організаціями та профспілками, оскільки їх вплив на працівників організації дуже велике;

- охорона. Передбачає не тільки контроль ситуації на робочих місцях, а й облік індивідуальних запитів працівників щодо умов праці [26];

- соціальне забезпечення співробітників. Крім обов'язкових відрахувань до державні позабюджетні фонди включає в себе організацію умов побуту працівників, наприклад, місць в дитячих садах, юридичної допомоги,

транспортного обслуговування та інші послуги, надання або оплату житла і т.д. [7];

- управління організаційної культурою. Ця функція забезпечує поділ працівниками етичних норм, цілей і цінностей організації, сприяє розвитку між співробітниками відносин взаємної поваги, доброзичливості, співробітництва і взаємної підтримки і т.д .;

- забезпечення позитивного ставлення до організації, її позитивного сприйняття споживачами, громадськістю та органами влади.

Функції управління персоналом узагальнені на рис. 2.4.

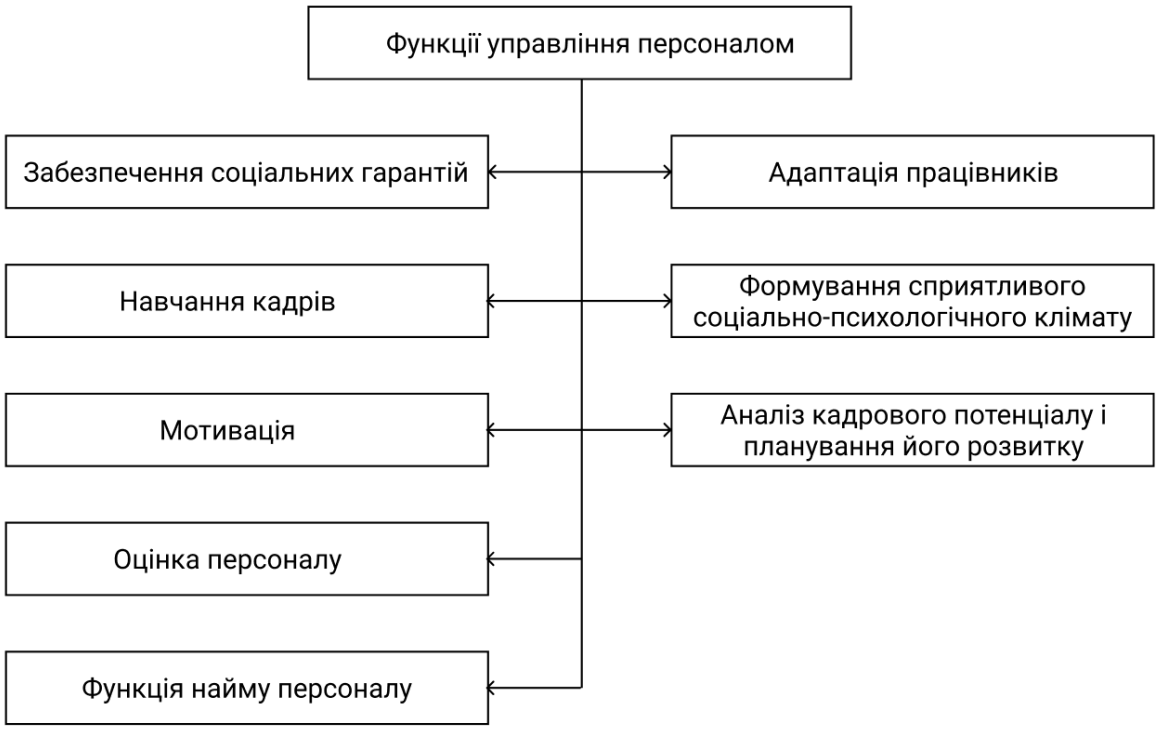

Рисунок 2.4 - Функції системи управління персоналом в організації

Перелік функцій управління персоналом може включати в себе і інші напрямки [4].

Таким чином, управління персоналом спрямоване на досягнення цілей організації і цілей працівника. З точки зору працівника компанії необхідне створення таких умов, в яких формується задоволеність працею у взаємозв'язку з виконанням працівником трудових обов'язків як умови досягнення цілей

організації. Організація орієнтується на отримання прибутку, тому система управління персоналом повинна забезпечувати можливості використання працівників для досягнення цієї мети, а також створювати умови для підвищення ефективності використання трудового потенціалу на перспективу.

2.2 Система управління персоналом в організації

Управління персоналом належить до найважливіших аспектів діяльності організації, оскільки тільки за умови ефективного використання трудових ресурсів в діяльності організації можливе здійснення беззбиткової діяльності формування прибутку. Саме тому управління персоналом може розглядатися як окремий аспект діяльності організації та як складова менеджменту компанії. Виділення питань кадрової політики в окремий елемент управління організацією пов'язано з особливою значущістю працівників як ресурсу ведення бізнесу. Взаємозв'язок управління персоналом із загальними питаннями управління проявляється через участь працівників у всіх інших аспектах діяльності організації [3].

Тому система управління персоналом як сукупність взаємопов'язаних елементів, за рахунок яких досягається найбільш ефективне використання трудових ресурсів на перспективу з точки зору отримання організацією прибутку і досягнення інших цілей управління діяльністю може розумітися як відособлена система управління або як частина менеджменту.

Організація діє на ринку для отримання прибутку, оскільки отримання прибутку обумовлене створенням певної продукції, потрібне об'єднання ресурсів, найбільш значущим з яких виступають людські ресурси. Як наслідок, потреба в залученні людських ресурсів знаходиться у взаємозв'язку з отриманням організацією прибутку [39]. Тому систему управління персоналом можна визначити як сукупність взаємопов'язаних елементів, за рахунок яких досягається найбільш ефективне використання трудових ресурсів на

перспективу з точки зору отримання організацією прибутку і досягнення інших цілей управління діяльністю організації.

У складі системи управління персоналом можна виділити окремі елементи, що представляють собою найбільш значущі складові даної системи, що забезпечують, в тому числі, її взаємозв'язок з іншими елементами управління в організації.

Елементи системи управління персоналом представлені на рис. 2.5.

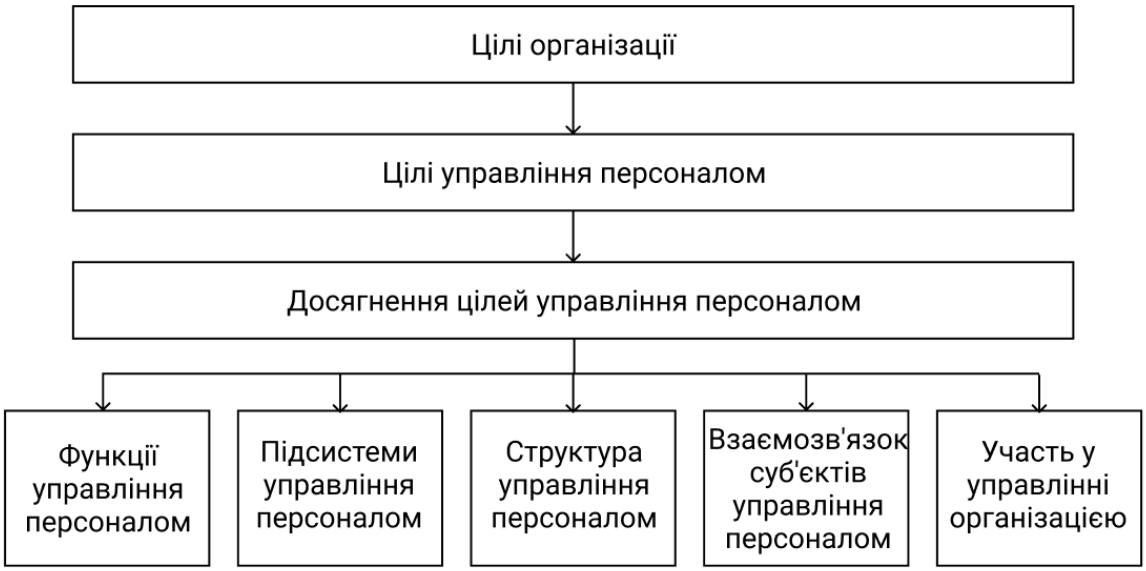

Рисунок 2.5 - Елементи системи управління персоналом організації

Елементи системи управління персоналом включають в себе цілі, функції, підсистеми, структуру управління персоналом, а тако взаємозв'язок суб'єктів управління персоналом і участь працівників в управлінні організацією.

Участь працівників в управлінні організацією можливо з точки зору участі у виробленні рішення або тільки у виконанні цих рішень, але, в будь-якому випадку, повинні бути сформовані ті механізми, якими забезпечується досягнення взаємозв'язку між цілями організації і цілями управління персоналом. Ці механізми представлені, в першу чергу, участю працівників в управлінні, а на рівні системи управління персоналом можуть бути реалізовані як через підсистему стимулювання праці персоналу, так і через інші підсистеми управління персоналом [16].

Функції управління персоналом можуть бути визначені виходячи з сукупності взаємопов'язаних між собою завдань, що забезпечують досягнення загальних цілей в управлінні персоналом. Ці функції відносяться до самої організації і до працівників. Для організації основною метою в управлінні персоналом є отримання прибутку, оскільки працівник повинен бути залучений в досягнення даної мети, формується широкий перелік функцій, пов'язаних з управлінням персоналом організації. Оскільки реалізація функцій управління персоналом може забезпечуватися в рамках певних сторін діяльності організації, пов'язаних з управлінням персоналом, окремо слід виділяти підсистеми управління персоналом, які забезпечують виконання функцій.

Підсистема управління персоналом являє собою сукупність методів, цілей і суб'єктів управління персоналом, що відповідають за реалізацію певної функції в управлінні працівниками.

Оскільки один суб'єкт управління персоналом може виконувати кілька функцій, крім того, існують керівники організації, які так чи інакше беруть участь в реалізації всіх основних функцій управління персоналом, слід говорити про структуру управління персоналом. Вона включає в себе керівників організації, в першу чергу, керівників, які здійснюють безпосередньо управлінський вплив по відношенню до певних працівникам для досягнення цілей, пов'язаних з управлінням персоналом [8].

Крім того, в структуру управління персоналом входять спеціалізовані суб'єкти, на яких покладено виконання окремих функцій виконання персоналом, при цьому дані співробітники спеціалізуються тільки на питаннях управління персоналом організації, надають підтримку керівникам. Як правило, спеціалізовані суб'єкти об'єднані в кадрову службу організації, яка додатково забезпечує ведення кадрової роботи.

Суб'єкти управління персоналом взаємопов'язані між собою, оскільки повинні бути орієнтовані на досягнення загальних цілей в управлінні персоналом, а керівники також орієнтуються на досягнення цілей підрозділів, які очолюють.

В результати формуються зв'язку між суб'єктами управління персоналом компанії [3].

Таким чином, систему управління персоналом можна визначити як сукупність взаємопов'язаних елементів, за рахунок яких досягається найбільш ефективне використання трудових ресурсів на перспективу з точки зору отримання організацією прибутку і досягнення інших цілей управління діяльністю організації. Елементи системи управління персоналом включають в себе цілі, функції, підсистеми, структуру управління персоналом, а також взаємозв'язок суб'єктів управління персоналом і участь працівників в управлінні організацією.

Управління персоналом як одна з найбільш значущих складових системи управління організацією в цілому активно розвивається в практиці російських і зарубіжних компаній. Це пов'язано з високим економічним ефектом, який створюють для компанії сучасні рішення, що дозволяють більш ефективно управляти персоналом. Найбільше значення, безумовно, має управління людськими ресурсами. Проте, також можна виділити ряд окремих технологій в управлінні персоналом, які охоплюють всі основні елементи, складові систему управління персоналом організації.

Найбільше значення мають сучасні рішення в області мотивації праці працівників. Основною тенденцією в удосконаленні мотивації праці персоналу слід вважати більш високу увагу, яку приділяють використання негрошових і нематеріальних винагород.

Отже, в плані негрошових винагород в зарубіжній практиці активно використовуються винагороди, пов'язані з громадським статусом працівника через оплачувані представницькі витрати, створюються пенсійні плани, консультативними службами забезпечується захищеність співробітників компанії, організовуються корпоративні заняття спортом.

Крім негрошового стимулювання, активно використовується в ряді зарубіжних компаній, також можна відзначити розвиток методів

нематеріального заохочення працівників. Окремі приклади використання методів нематеріального стимулювання праці співробітників компаній представлені в табл. 2.3 [1].

Таблиця 2.3 - Сучасні технології нематеріального стимулювання праці

Технологія	Опис
Представницькі витрати	Використовуються, в основному, за кордоном. Працівник у відрядженні розміщується в престижному готелі, навіть простий фахівець може бути розміщений в номері «люкс». Для організації це питання престижу компанії, оскільки вона демонструє конкурентоспроможність бізнесу. З точки зору працівника можливість проживання в номері, зазвичай призначеному для найбільш заможних гостей, є не тільки стимулом кращим чином виконати обов'язки в період відрядження, а й діяти найефективніше в подальшому. Крім того, ряд компаній за кордоном став надавати працівникам корпоративні карти з високим статусом, тому працівник має можливість розрахунку карткою, яка сама по собі визначає високий статус у суспільстві. Слід враховувати, що за кордоном ставлення до престижних картками інше, ніж в Україні, вони визначають становище людини в суспільстві
Пенсійні плани	Призначені для найбільш цінних працівників організації. Заключаются в можливості дострокового виходу на пенсію за умови виконання встановлених працівнику кар'єрних перспектив.

	Даний інструмент негрошового стимулювання пов'язаний з приватними пенсійними фондами, використовується і в Україні
Консультативні служби	Практикуються, в основному, великими іноземними компаніями. Консультативна служба забезпечує вирішення будь-яких побутових питань працівника, починаючи від факту затримання співробітниками правоохоронних органів за дрібне правопорушення і закінчуючи необхідністю проведення невеликого ремонту у працівника будинку. Основним фактором виступає високий рівень захищеності, який пропонує працівникові організація
Заняття спортом	На додаток до став вже звичними клубними картками, які використовуються і в Україні, за кордоном стали практикуватися командні тренування. Наймається корпоративний тренер, який проводить заняття в спортзалі організації. Ціль командної тренування полягає не тільки в заняттях спортом, але і в координації зусиль працівників компанії. Тим самим цей інструмент сприяє кращому соціально-психологічного клімату в колективі

Отже, серед нематеріальних стимулів особливе місце займає організація робочого місця як спосіб заохочення інших працівників до більш продуктивної праці, крім того, використовується стимулювання вільним часом, ряд компаній пропонує можливість вибору відпустки, сплаченого компанією як поєднання негрошових і нематеріальних стимулів. Інакше кажучи, в зарубіжній практиці

управління персоналом значна увага приділяється підвищенню ефективності здійснення витрат на стимулювання праці працівників.

Сучасні технології використовуються і в інших аспектах управління персоналом. Зокрема, в навчанні персоналу все більш активно використовуються технології віддаленого навчання, наприклад при створенні груп в соціальній мережі компанії, де розміщуються короткі навчальні ролики, призначені для працівників конкретного підрозділу [17]. Це рішення також дозволяє знизити витрати на управління персоналом. Таким чином, в плані негрошових винагород в зарубіжній практиці активно використовуються винагороди, пов'язані з громадським статусом працівника через оплачувані представницькі витрати, створюються пенсійні плани, консультативними службами забезпечується захищеність співробітників компанії, організовуються корпоративні заняття спортом. Серед нематеріальних стимулів особливе місце займає організація робочого місця як спосіб заохочення інших працівників до більш продуктивної праці, крім того, використовується стимулювання вільним часом, ряд компаній пропонує можливість вибору відпустки, сплаченого компанією як поєднання негрошових і нематеріальних стимулів.

2.3 Ефективність системи управління персоналом в організації

Правильне управління персоналом в організації є ключовим фактором успіху фірми, тому що саме персонал, його мотивація і зацікавленість в роботі підприємства є тим двигуном, який може посилити використання наявних ресурсів, або може знищитинайперспективніші і багатообіцяючі плани організації.

Завдання, пов'язані з управлінням персоналу, виникають на всіх функціональних ділянках і рівнях управління економічної діяльністю. При їх вирішенні слід виходити з того, що в центрі уваги варто працююча людина, незалежно від того, яке місце він займає в організації і які завдання перед ним виникають. Робота з персоналом, останнім часом, виступає не як управління,

засноване лише на адміністративних методах, а являє собою реалізацію тих управлінських рішень, які засновані на узгодженні системи інтересів людини і підприємства [3].

Розгляд даної теми є актуальним в даний час, так як ефективність управління співробітниками, а значить, і вся ефективність діяльності організації в цілому, залежить від того, наскільки цілі підприємства і цілі його працівників збігатимуться між собою [4].

В даний час немає єдиного підходу до визначення критеріїв ефективності в управлінні персоналом і до проблеми вимірювання ефективності трудової діяльності. Складність полягає в тому, що процес трудової діяльності співробітників пов'язаний з виробничим процесом його кінцевими результатами, соціальною діяльністю суспільства, а також економічним розвитком підприємства.

Таблиця 2.4 - Загальні підходи до визначення критеріїв ефективності управління персоналом

Підхід	Сутність підходу
1. Підхід «А»	Сутність даного підходу полягає в тому, що кінцеві результатипроізводства повинні служити критеріальними показниками ефективностіуправленія персоналом. Критеріальними показниками виступають: а) прибуток підприємства; б) витрати на 1 рубль продукції або послуги; в) рівень рентабельності; г) дивіденди, в розрахунку на 1 акцію
2. Підхід «Б»	Даний підхід передбачає, що результативність і складність «живого праці» повинні відображати такі критеріальні показники: а) продуктивність праці;

	б) загальний ФОП; в) темпи зростання продуктивності праці і заробітної плати; г) питома вага заробітної плати в собівартості продукції.
3. Підхід «В»	На підставі підходу «В» ефективність управління персоналом визначається організацією і мотивацією праці, психологічним станом колективу. Критеріальні показники, які виділяються даного підходу: а) плинність кадрів; б) рівень кваліфікації; в) витрати на навчання співробітників; г) витрати на створення реалізацію соціальних програм.

Оцінка результативності праці кожного працівника спрямована на визначення рівня ефективності виконання його роботи. [7] Оцінка праці персоналу переслідує здійснення певних цілей, таких як:

По - перше, підвищення ефективності роботи співробітників.

По - друге, призначення справедливого і гідної винагороди за виконану роботу.

По - третє, прийняття рішення, пов'язаного з трудовою кар'єрою співробітника.

Творчі здібності людини виявляються через його компетенцію, яка є ключовим поняттям всієї концепції управління персоналом.

Концепція - це доцільне поєднання здібностей, особистісних якостей і мотивації персоналу фірми, які розглядаються в тимчасовому інтервалі.

Оцінюючи рівень компетентності для персоналу, розглядаються в найзагальнішому вигляді:

а) здатності персоналу (тобто рівень освіти, обсяг знань, професійні навички і досвід роботи в певній сфері);

б) особистісні властивості (тобто ініціативність, комунікабельність, надійність, працездатність, пунктуальність, відданість роботі та стресостійкість);

в) мотивацію (тобто коло професійних і особистих інтересів прагнення зробити кар'єру, досягти поставлених перед собою цілей).

Компетенція персоналу є категорією дуже динамічною. Вона може бути значно підвищена за рахунок постійного навчання, самоосвіти, прийому працівників з боку і особливо мотивації. Тому важливим завданням в управлінні персоналом є поліпшення результативності його дій, яка залежить від компетенції і мотиваційного фактора.

При оцінці результативності праці важливе значення має бездоганне і точне визначення кількісних і якісних показників, які відображають кінцеві цілі підрозділи або фірми в цілому. Показники, за якими оцінюються працівники, називаються критеріями оцінки. [12] Концепція оцінки праці персоналу фірми грунтується на тому, що в якості критеріїв оцінки беруться показники, оцінюють рівень компетенції працівників. [16]

Дослідження, проведені в декількох фірмах США, показали, що частота використання різних критеріїв становить:

- якість виконаної роботи - 94%;

- обсяг наданої роботи - 91%;

- знання виконуваного справи- 86%.

Особистими якостями, які використовуються в ролі до - надійність і відповідальність - 87%.

Вибір критеріїв визначається призначенням критеріїв оцінки, тобто тим, для вирішення яких завдань використовуються результати оцінки. Так, наприклад, якщо основною метою є підвищення результатів роботи і визначення заробітної плати працівників, то оцінку здійснюють за критеріями результативності. Якщо ж метою є кар'єрне просування, то застосовуються критерії, які визначають потенційну результативність на новому місці роботи. [21]

Особистими якостями, які використовуються в ролі критеріїв виступили:

- ініціативність і підприємливість - 88%;

- комунікабельність і контактність - 88%;

- надійність і відповідальність - 87%.

Вибір критеріїв визначається призначенням критеріїв оцінки, тобто тим, для вирішення яких завдань використовуються результати оцінки. Так, наприклад, якщо основною метою є підвищення результатів роботи і визначення заробітної плати працівників, то оцінку здійснюють за критеріями результативності. Якщо ж метою є кар'єрне просування, то застосовуються критерії, які визначають потенційну результативність на новому місці роботи. [21]

Оцінка результативності праці працівника спрямована на реалізацію основних цілей:

1. Адміністративні цілі.

2. Інформаційні цілі.

3. Мотиваційні цілі.

Ефективність управління персоналом здійснюється на основі таких методів:

по - перше, методу оцінки досягнення цілей;

по - друге, методу оцінки компетенцій;

по - третє, методу мотивації;

в - четверте, методу вивчення статистики людських ресурсів;

по - п'яте, методу оцінки витрат.

Щоб вистояти в нинішній економічній ситуації, підприємствам потрібно зосередитися на персоналі. За свою частку на ринку доведетьс боротися, втілюючи нові бізнес - ідеї, що неможливо без гідного кадрового резерву.

Дослідження BCG і Всесвітньої федерації асоціацій управління персоналом показало, що саме сильний відділ кадрової служби багато в чому є запорукою успіху будь-якого бізнесу. Дослідники проаналізували фондові

показники «Кращих роботодавців» Forbes за 20 років, і зіставили їх з даними індексу Standard & Poor's 500.

Підсумки дослідження такі - результати 100 організацій з найбільш ефективною системою управління персоналом виявилися вищими індексу на 100%. Це однозначно показує пряму залежність фінансових показників компанії від ефективного управління персоналом в ній.

Фактично, в даний час відбувається перетворення в області управління людськими ресурсами. Змінюються завдання і цілі кадрової роботи - вона перестає бути чисто адміністративної. Менеджер відповідає за все, що відбувається в компанії в сфері управління персоналом, саме він формує корпоративну культуру, створює імідж компанії. На Заході від менеджера потрібне знання всіх бізнес-процесів компанії, від виробництва до маркетингової діяльності.

На чолі концепції управління персоналом варто людський потенціал. Кадрова робота, яка часто складається як виключно адміністративна, стримує перспективне управління персоналом. [17]

Стратегічне ж управління дозволяє оперативно реагувати на виклики із зовнішнього середовища, а також має на увазі гнучкість і орієнтацію на клієнта. Кадрова політика при цьому ставить собі за мету повне задоволення запитів кожного співробітника для розкриття його особистого потенціалу.

Таким чином, можна прийти до висновку про те, що оцінка ступеня результативності праці кожного працівника - обов'язковий елемент контролю будь-якої фірми. [24] Це одна з ключових функцій управління персоналом, спрямована на визначення рівня ефективності виконання поставлених завдань.

У загальному вигляді оцінка результативності праці працівника повинна включати в себе виконання наступних заходів:

1. Чітку формулювання вимог, що пред'являються до конкретної посади.

2. Формування системи критеріїв оцінки рівня компетенції працівника, орієнтовану на виконання певних посадових вимог.

3. Комплексну оцінку праці працівника.

4. Створення механізму, що зв'язує результати оцінки праці працівника з системою винагороди за працю, тобто з визначення заробітної плати, розміру премій та пільг.

5. Створення механізму, що зв'язує результати оцінки праці працівника з системою підвищення кваліфікації та перепідготовки кожног з співробітників.

2.4 Висновок по розділу

Систему управління персоналом можна визначити як сукупність взаємопов’язаних елементів, завдяки яким у майбутньому досягається найбільш ефективне використання трудових ресурсів з точки зору організації прибутку та досягнення інших цілей управління діяльність організації.

Елементи системи управління персоналом включають цілі, функції, підсистеми, структуру управління персоналом, а також взаємозв'язок суб'єктів управління персоналом та участь працівників в управлінні організацією.

Управління персоналом спрямоване на досягнення цілей організації та цілей працівника. З точки зору працівника підприємства необхідно створити такі умови, при яких формується задоволеність роботою у зв'язку з виконанням працівником трудових обов'язків, як умови досягти цілей організації.

Організація орієнтована на прибуток, тому система управління персоналом повинна надавати можливості використовувати працівників для досягнення цього, а також створювати умови для підвищення ефективності трудового потенціалу в майбутньому.

3 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Моделювання та аналіз програмного забезпечення

Розроблена система може виконуватися на операційних системах Linux та macOS. Робота з Microsoft Windows також можлива, але не була протестована.

Браузерна частина була розроблена за допомогою мови Javascript, з використанням бібліотек React та Redux. Використаний паттерн MVC та дозволяє швидко та зручно додавати новий функціонал та відокремлювати різні частини коду, так як MVC дозволяє відокремити дані від логіки відображення чи від бізнес-логіки, Взаємодія з серверними частинами відбувається по протоколу HTTP REST з кодуванням даних JSON.

Серверна частина також була розроблена на Javasctipt, з використанням фреймворку ExpressJS. Чат був розроблений за допомогою Web-socket. Авторизація за допомогою JWT token. Опис інтерфейсів наведено у таблиці 3.1, опис функцій – у таблиці 3.2., опис файлів наведено у таблиці 3.3.

Таблиця 3.1 – Опис інтерфейсів системи

Інтерфейс	**Опис**
LandingPage	Інтерфейс з можливістю вибору типу користувача для подальшої авторизації
ErrorMsg	Інтерфейс з сповіщеням про помилку
Admin/AdminLogin	Інтерфейс авторизації для адміністратора
Admin/Sidenav	Інтерфейс меню для адміністратора
Admin/Contact/AddForm	Інтерфейс для адміністратора з можливістю додавання нового кліента
Admin/Contact/Contact	Інтерфейс для адміністратора з детальною інформацією про кліента
Admin/Contact/Contacts	Інтерфейс для адміністратора з усіма кліентами
Admin/Contact/EditContact	Інтерфейс для адміністратора з можливостю редагування кліента
Admin/Leads/AddFormLead	Інтерфейс для адміністратора з можливістю додавання нового керівника
Admin/Leads/EditLead	Інтерфейс для адміністратора з можливістю редагування керівника
Admin/Leads/Lead	Інтерфейс для адміністратора з детальною інформацією про керівника
Admin/Leads/Leads	Інтерфейс для адміністратора з усіма керівниками

Admin/ServiceRequest/AddForm	Інтерфейс для адміністратора з можливістю додавання нової задачи
Admin/ServiceRequest/EditService	Інтерфейс для адміністратора з можливостю редагування задачи
Admin/ServiceRequest/Service	Інтерфейс для адміністратора з усіма задачами
Admin/ServiceRequest/ServiceRequest	Інтерфейс для адміністратора з детальною інформацією про задачу
Admin/templates/LoaderTemplate	Інтерфейс загрузки адміністратора
Admin/templates/TitleTemplate	Інтерфейс кнопки add
Admin/User/AddUser	Інтерфейс для адміністратора з можливістю додавання нового співробітника
Admin/User/AllUser	Інтерфейс для адміністратора з усіма співробітниками
Admin/User/DeleteUser	Інтерфейс для адміністратора з можливістю видалення співробітника
Employee/EmployeeLogin	Інтерфейс авторизації для співробітника
Employee/Sidenav	Інтерфейс меню для співробітника
Employee/Contact/Contact	Інтерфейс для співробітника з детальною інформацією про кліента
Employee/Contact/Contacts	Інтерфейс для співробітника з усіма кліентами
Employee/Leads/Lead	Інтерфейс для співробітника з детальною інформацією про керівника

Employee/Leads/Leads	Інтерфейс для співробітника з усіма керівниками
Employee/ServiceRequest/Service	Інтерфейс для співробітника з усіма задачами
Employee/ServiceRequest/ServiceRequest	Інтерфейс для співробітника з детальною інформацією про задачу
Employee/templates/LoaderTemplate	Інтерфейс загрузки співробітника
Manager/ManagerLogin	Інтерфейс авторизації для менеджера
Manager/Sidenav	Інтерфейс меню для менеджера
Manager/Contact/AddForm	Інтерфейс для менеджера з можливістю додавання нового кліента
Manager/Contact/Contact	Інтерфейс для менеджера з детальною інформацією про кліента
Manager/Contact/Contacts	Інтерфейс для менеджера з усіма кліентами
Manager/Contact/EditContact	Інтерфейс для менеджера з можливостю редагування кліента
Manager/Leads/AddForm	Інтерфейс для менеджера з можливістю додавання нового керівника
Manager/Leads/EditLead	Інтерфейс для менеджера з можливістю редагування керівника
Manager/Leads/Lead	Інтерфейс для менеджера з детальною інформацією про керівника
Manager/Leads/Leads	Інтерфейс для менеджера з усіма керівниками

Manager/ServiceRequest/AddForm	Інтерфейс для менеджера з можливістю додавання нової задачи
Manager/ServiceRequest/EditService	Інтерфейс для менеджера з можливостю редагування задачи
Manager/ServiceRequest/Service	Інтерфейс для менеджера з усіма задачами
Manager/ServiceRequest/ServiceRequest	Інтерфейс для менеджера з детальною інформацією про задачу
Manager/templates/LoaderTemplate	Інтерфейс загрузки менеджера

Таблиця 3.2 – Опис функцій системи

Функція	**Опис**
serviceRequestSchema	Функція для валідації даних serviceRequest
router.post	Функція, яка описує логіку методу створення serviceRequest
router.get	Функція, яка описує логіку методу отримання serviceRequest
delete	Функція, яка видаляє serviceRequest
put	Функція, яка змінює serviceRequest
leadSchema	Функція для валідації даних керівників
contactSchema	Функція для валідації даних клієнтів
registerSchema	Функція для валідації даних реєстрації
loginSchema	Функція для валідації даних авторизації
adminverify	Функція перевірки авторизаціх користувача, а саме адміністратора

router.get(“/servicerequest”)	Функція отримання даних servicerequest
router.get(“/lead”)	Функція отримання даних керівників
router.get(“/contact”)	Функція отримання даних кліентів
employeeauth	Функції режстрації та авторизації
employeeverify	Функція перевірки авторизаціх користувача, а саме співробітника

Таблиця 3.3 – Опис файлів системи

Файл	**Опис**
Contact.js	Моделі бази даних для клієнтів
Lead.js	Моделі бази даних для керівників
ServiceRequest.js	Моделі бази даних для задач
User.js	Моделі бази даних для користувачів
Adminauth.js	Логіка авторізації адміністратора
AdminDashboard.js	Логіка основних методів адміністратора
Adminverify.js	Перевірка авторизації адміністратора
Employeeauth.js	Логіка авторізації співробітника
EmployeeDashboard.js	Логіка основних методів співробітника
Employeeverify.js	Перевірка авторизації співробітника
Managerauth.js	Логіка авторізації менеджера
ManagerDashboard.js	Логіка основних методів менеджера
Managerverify.js	Перевірка авторизації менеджера
Index.js	Налаштування шляхів АПІ
App.js	Головний файл фронтенду
serviceWorker.js	Фоновий процес фронтенду

3.2 Аналіз безпеки даних

Користувацькі паролі захищено за допомогою bcrypt, що майже знеможливлює перебор паролю та пошук праобразу навіть за умови доступу до його хешу. Алгоритм bcrypt - це алгоритм хешування, який масштабується за допомогою апаратного забезпечення (через настроювання кількісті раундів). Його повільність і кілька раундів гарантує, що зловмисник повинен розгорнути величезні кошти і обладнання, щоб зламати ваші паролі. Алгоритм bcrypt використовує алгоритм Eksblowfish для хеш-паролів. Хоча фаза шифрування Eksblowfish і Blowfish абсолютно однакова, ключова фаза розкладу Eksblowfish гарантує, що будь-яка наступний стан залежить як від солі, так і від ключа (пароль користувача), і ніякий стан не може бути попередньо обчислено без знання обох. Через це ключового відмінності bcrypt є одностороннім алгоритмом хешування. Доступ до БД наявний лише у серверної частини та в обмеженому обсязі, за допомогою налаштувань бази та інфраструктури, що майже знеможливлюють зловмисний доступ та дозволяють швидко його заблокувати. Повідомлення у web-socket також зашифровані.

3.3 База даних

У якості бази даних для вебзастосунку було обрано MongoDB через простоту використання та простоту розробки. MongoDB - це нереляційна система управління базами даних (СУБД) із відкритим кодом, яка використовує гнучкі документи замість таблиць і рядків для обробки та зберігання різних форм даних. Як рішення NoSQL, MongoDB не вимагає реляційної системи управління базами даних, тому вона забезпечує еластичну модель зберігання даних, яка дозволяє користувачам легко зберігати та запитувати багатовимірні типи даних. Це не тільки спрощує управління базами даних для розробників, а й створює високомасштабоване середовище для крос-платформних додатків та служб.

Документи MongoDB або колекції документів є основними одиницями даних. Відформатовані як бінарний JSON, ці документи можуть зберігати різні

типи даних і розподілятися між кількома системами. Оскільки MongoDB використовує динамічний дизайн схеми, користувачі мають неперевершену гнучкість при створенні записів даних, запитах до колекцій документів за допомогою агрегування MongoDB та аналізі великих обсягів інформації.

Документно-орієнтована БД працює дуже швидко, а використання Mongoose дозволяє взагалі не використовувати запити Mongo, окрім складних запитів, що максимально полегшує розробку слою роботи з базою даних. Використання індексів дозволяє прискорити вибірку з бази. Усього у системі використовується 5 таблиці. Опис структури БД наведено у наступній таблиці.

Таблиця 4.4 – Структура БД

Таблиця	**Назва колонки**	**Тип**	**Опис**
contacts	id	objectId	Унікальний ідентифікатор контакту
contacts	date	varchar(128)	Дата створення контакту
contacts	title	varchar(128)	Опис контакту
contacts	email		Електронна адреса контакту
contacts	number		Номер телефону контакту
contacts	address		Адреса контакту

Продовження таблиці 2.4

Таблиця	**Назва колонки**	**Тип**	**Опис**
contacts	__v		Версія документу контакту

leads	id		Унікальний ідентифікатор керівника
leads	department		Опис від
leads	title		
leads	client		
leads	number		
leads	status		
leads	__v		

Продовження таблиці 2.4

Таблиця	**Назва колонки**	**Тип**	**Опис**
servicerequests	id		
servicerequests	date		
servicerequests	title		
servicerequests	client		
servicerequests	manager		
servicerequests	expected_revenue		
servicerequests	probability		
servicerequests	status		
servicerequests	expected_closing		
servicerequests	priority		
servicerequests	__v		
users	id		
users	date		
	fname		
users	lname		
users	email		

users	password		
users	type		
users	__v		
message	id_g		
message	id_s		
message	cheked		
message	content		
message	date		

На рисунку 3.1 наведено діаграму бази даних.

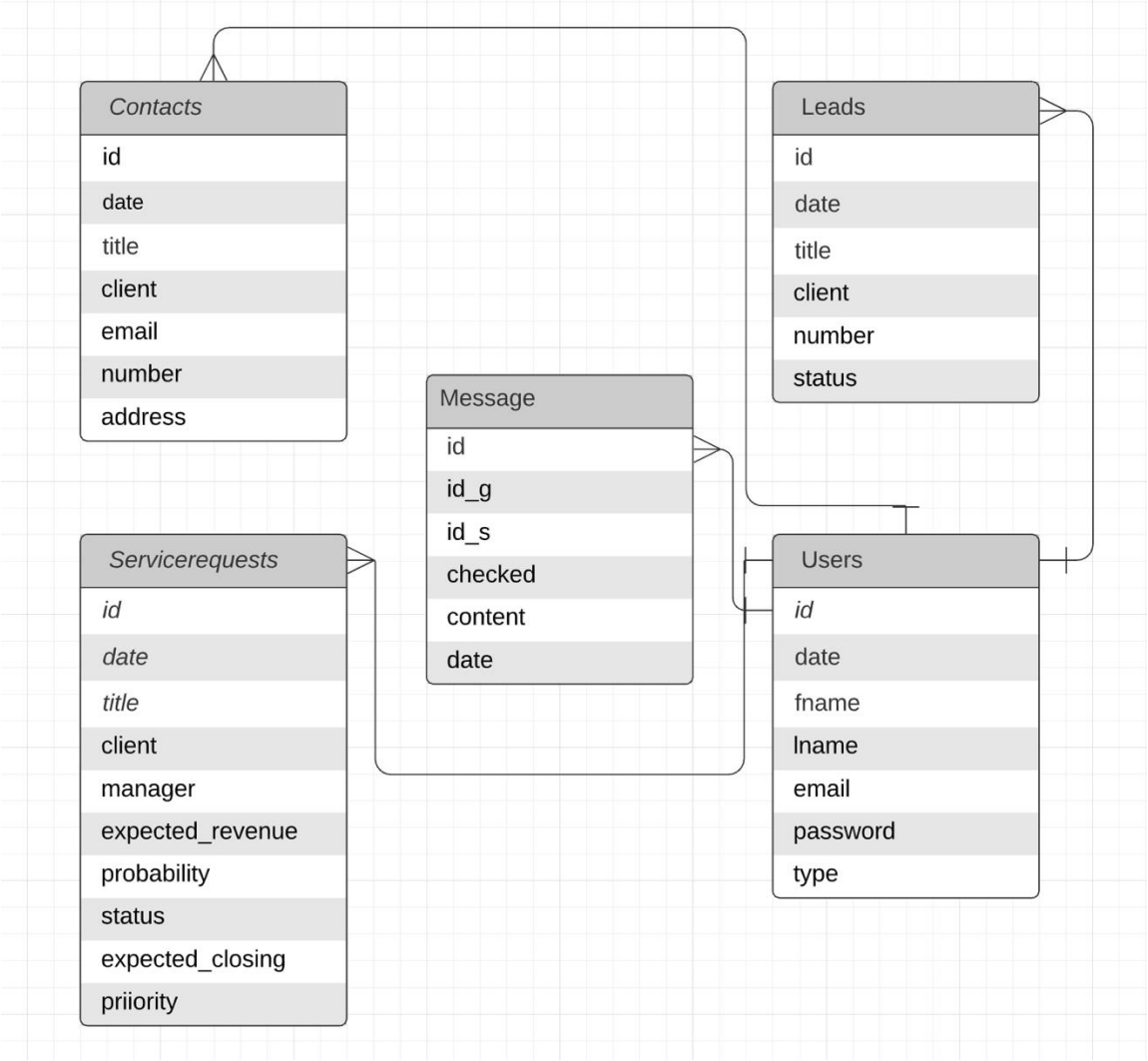

Рисунок 3.1 – Діаграма бази даних

3.4 Архітектура вебзастосунку

Вебзастосунок побудований за сервісною архітектурою. Вона складається з двох основних частин – браузерний додаток (фронтенд) та API сервер. Використання браузерного додатку дозволяє мати доступ до системи майже з будь-якого пристрою, що має доступ до Web, а використання окремого API серверу дозволяє у майбутньому створити клієнти під будь-яку платформу (Windows, Linux, Android, iOS, macOS і т.д.) без необхідності вносити зміни у інші компоненти системи. Усі компоненти системи можна масштабувати в залежності від навантаження, а проблеми з одним компонентом не матимуть вплив на роботу іншого.

3.4.1 Браузерний додаток (фронтенд)

Браузерна частина написана на мові Javascript, з використанням бібліотек React та Redux. React - це JavaScript бібліотека GUI з відкритим вихідним кодом, зосереджена на одній конкретної мети - ефективному виконання завдань в рамках розробки призначеного для користувача інтерфейсу. Його можна віднести до категорії "V" в архітектурному шаблоні MVC (модель-вид-контролер). React дозволяє повторно використовувати компоненти, які були розроблені в інших додатках, що використовують ту ж функцію. Можливість повторного використання компонента є явною перевагою для розробників. Компонент React створити простіше, оскільки він використовує JSX, опциональное розширення синтаксису JavaScript, яке дозволяє комбінувати HTML з JavaScript.

JSX – це суміш JavaScript і HTML. Воно робить весь процес написання структури сайту зрозумілішим. Крім того, розширення також значно спрощує рендеринг декількох функцій. Хоча JSX може бути не найпопулярнішим розширенням синтаксису, воно довело свою ефективність при розробці спеціальних компонентів або додатків великого обсягу.

Redux - це контейнер з передбачуваним станом для додатків JavaScript і дуже цінний інструмент для управління станом додатки. Також це популярна бібліотека для управління станом в додатках React, але її можна використовувати і з Angular, Vue.js і зі старим добрим JavaScript.

Головна складність Redux полягає в тому, що багато розробників не розуміють, коли його використовувати. Чим більше і складніше стає додаток, тим вище ймовірність того, що Redux буде вам корисний. Якщо ви починаєте працювати над додатком і очікуєте, що в недалекому майбутньому воно істотно виросте, ви можете відразу використовувати Redux: так у міру зміни і масштабування програми ви зможете легко впроваджувати все нові функції, уникаючи рефакторінга великої кількості готового коду.

Він дозволяє створювати SPA (Single page application), тобто односторінкові додатки, які змінюють свій стан без переходу на іншу сторінку, завдяки чому досягається висока швидкість роботи та простота розробки. Завдяки webpack вихідний код додатку обфусфікується, тому його буде складно змінити або викрасти. Також завдяки webpack досягається сумісність з великою кількістю браузерів за рахунок генерації поліфілів – реалізації вбудованих методів Javascript, які не існували у попередніх версіях браузерів. Webpack- це статичний модульний збирач для додатків на JavaScript. Програми, написані на JavaScript, постійно ускладнюються, тому для збору модулів все частіше використовують спеціальний інструмент - Бандлер. Подібні інструменти дозволяють розробникам упаковувати, компілювати і в цілому організовувати всі ресурси, необхідні для проекту. Можна використовувати не тільки сторонні бібліотеки, а й власні файли. Подібна модульна система дозволяє домогтися кращої організації проекту, так як він розбивається на невеликі модулі. Вебпак на даний момент є одним з найпотужніших подібних Бандлерів, тобто модульний збирачів. Він має відкритий вихідний код і дозволяє вирішувати безліч завдань. Як і інші інструменти розробника, вебпак має свої плюси і мінуси.

Почнемо з плюсів: він відмінно підходить для роботи з односторінкового додатками. Також вебпак може здійснювати просунутий поділ коду. Через ці та інші переваг він є одним з найбільш популярних інструментів JS-розробки на даний момент.

Мінуси: трохи складно розібратися в його роботі, частина документації застаріла через великої кількості змін в оновленнях.

Інструменти розробки React надають зручний спосіб запуску, відладки та побудови додатку. Завдяки популярності даного фреймворку, для нього існує багато стороніх бібліотек та компонентів, а також він активно підтримується та оновлюється. Завдяки використанню Javascript, код нетипізований, що пришвидшує його написання.

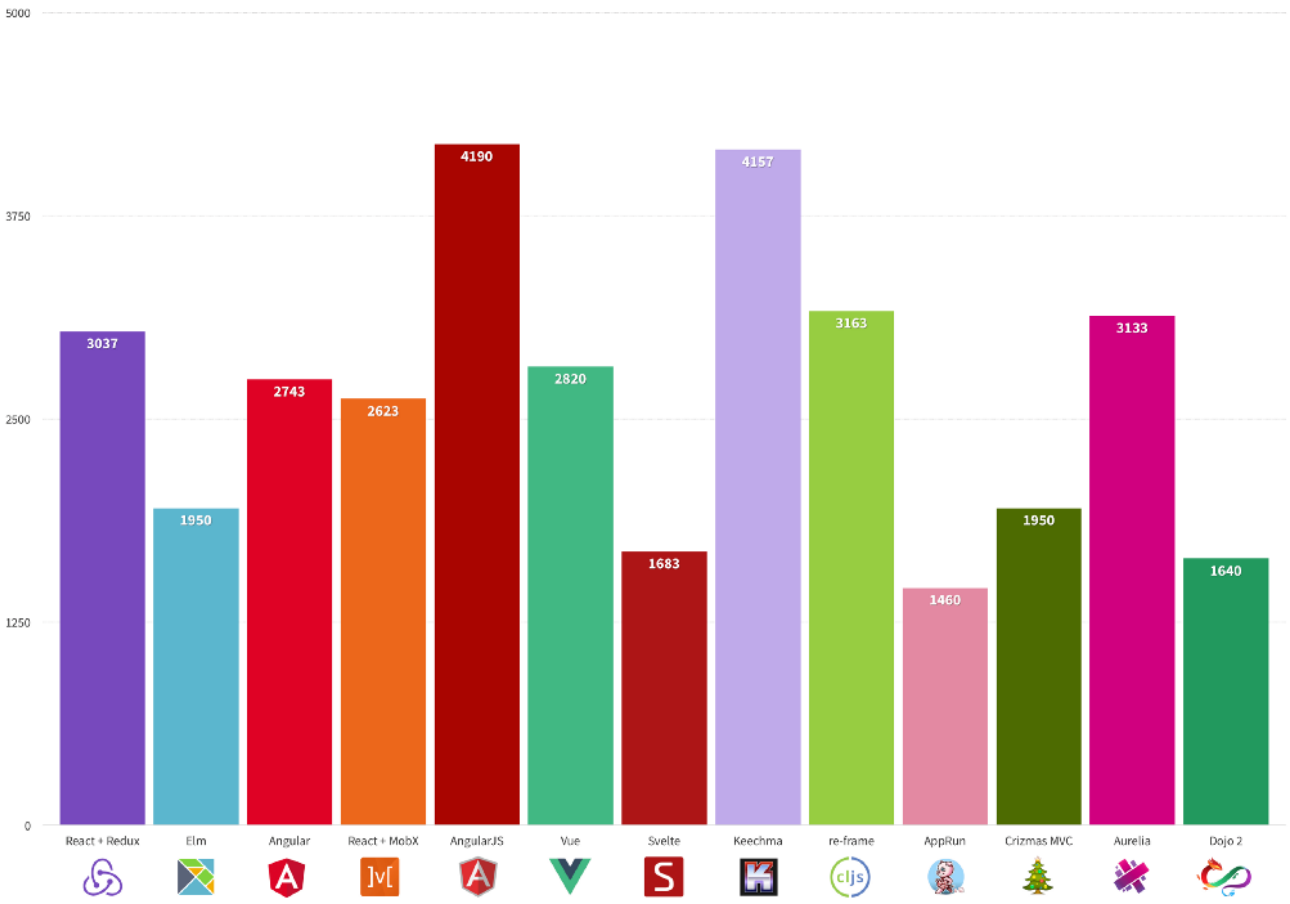

Рисунок 2.2 – Порівняння React+Redux з іншими популярними фреймворками

3.4.2 API сервер (бекенд)

API сервер розроблено з використанням мови Javascript, та фреймворку ExpressJS. ExpressJS – це фреймворк для веб-розробки для NodeJS - стандартний для більшості існуючих сьогодні додатків Node.js.

Основна ідея Node.js: використовувати неблокуючий подієво-орієнтований ввід / вивід, щоб залишатися легковажним і ефективним при поводженні з додатками, що обробляють великі обсяги даних в реальному часі і функціонуючими на розподілених пристроях. По суті, це означає, що Node.js не є платформою на всі випадки життя, яка буде домінувати в світі веб-розробки.

Навпаки, це платформа для вирішення строго визначених завдань. Розуміти це абсолютно необхідно. Зрозуміло, не варто використовувати Node.js для операцій, інтенсивно навантажують процесор, більш того - застосування Node.js в важких обчисленнях фактично анулює всі його переваги. Node.js дійсно хороший для створення швидких масштабованих мережних додатків, оскільки дозволяє одночасно обробляти величезну кількість з'єднань з великою пропускною здатністю, що рівноцінно високою масштабованості.

Тонкощі роботи Node.js «під капотом» досить цікаві. У порівнянні з традиційними варіантами веб-сервісів, де кожне з'єднання (запит) породжує новий потік, навантажуючи оперативну пам'ять системи і, врешті-решт, розбираючи цю пам'ять без залишку, Node.js набагато економічніше: він працює в єдиному потоці, при викликах використовує неблокірующій ввід / вивід, дозволяє підтримувати десятки тисяч конкурентних з'єднань. Цей фреймворк дозволяє без проблем створювати HTTP REST API сервери. API сервер взаємодіє з браузерним програмним забезпеченням через HTTP, інформація кодується за допомогою JSON, через це є можливість без проблем працювати з нею як програмно, так і користувач. API сервер обмінється інформацієї з базою даних для надання інформації, а також створення нових аналізів та регістрації користувачів. Завдяки тому, що Javasctipt використовуються, як на фронтенді, так

і на бекенд частині, полегшується розробка, через те, що використовується одна та єдина мова програмування.

3.5 Висновки по розділу

У даному розділі було розроблено архітектури інтелектуальної системи. Була обрано набір мов програмування та фрейморків що є найбільш ефективними та зручними розробки саме даного програмного забезпечення, а також операційні системи на яких впершу чергу буде перевірена роботоздатність. Описано паттерни, класи системи та їх класи. Також була проаналізована безпека даних у інтелектуальній системі.

4 РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

4.1 Ідея проекту

У даному розділі визначено можливості розроблюваного веб-застосунку по виходу на ринок та можливості конкуренції з іншими продуктами, що вже наявні наринку.

Цільовою аудиторією даної системи є компанії, що плануюсь розвиватися та використовувати crm-системи, чати, співробітники що взаємодіють між собою тощо.

Сутністю розробки є вебзастосунок для управління співробітниками, ведення статистика, та взаємодії співробітників між собою.

Призначенням проекту є надання можливості управління співробітниками та взаємодії між ними для будь-якої компанії.

Основною вигодою від даного продукту є можливість листування між співробітниками в середені сервісу, що надає велику перевагу серед конкурентів, а також ведення статистики співробітників, завтяку чому можна аналізувати розвиток компанії.

З метою отримання уявлення про ідею та можливі ринки збуту продукту, наведено детальний опис у таблиці 4.1

Таблиця 4.1 – Опис ідеї стартап проекту

Зміст ідеї	Напрямки застосування	Вигода для користувача
Розроблений веб-застосунок може бути використаний компаніями для автоматизації роботи з співробітниками, збирання даних для аналізу і поліпшення бізнес-процесів, а також для взаємодії співробітників.	IT компанії	Можливість листування між співробітниками в середені сервісу, що надає велику перевагу серед конкурентів, а також ведення статистики співробітників, завтяку чому можна аналізувати розвиток компанії.

З огляду на те, що такий вид роботи, як full-remote, стає все більш популярною в Україні та світі, в майбутньому може з'явитися більше можливостей та інтеграцій системи з іншими сервісами.

Наступним кроком розробки стартап проекту є аналіз техніко-економічних переваг ідеї у порівнянні з конкурентами. Результати даного аналізу наведено у таблиці 4.2.

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів			W (слабка сторона)	N (нейтральна сторона)	S стор
		Вебзастосунок	Teams	Jira			
1	Статистика співробітників	+	+	-	-	-	+
2	Можливість листування	+	-	-	-	-	+
3	Таймтрекер	+	+	+	-	+	-
4	Інтеграція з браузерами та іншим ПЗ	-	+	+	-	-	+

Виходячи з таблиці вище можна зазначити, що інтелектуальна система має певні переваги перед конкурентними продуктами, насамперед – можливіть спілкуватись з колегами за допомогою внутрішнього чату, що робить конкурентів абсолютно непригодними для призначення проекту. Також можна зазначити що конкуренти мають низку переваг, які було б вигідно реалізувати у вебзастосунку для підвищення зручності використання. Також можливе використання методів машинного навчання що на данному етапі розробки є неможливим через необхідність у дуже великій кількості даних.

4.2 Технічний аудит проекту

У даному розділі розглянуто методи реалізації ідеї стартап проекту. Зв’язок між технологіями реалізації та ідеями наведено у таблиці 4.3

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

Ідея проекту	Технології реалізації	Наявність технології	Доступність технології
Керування підлеглими	ExpressSJ+mongoDB	Наявна	Доступна безкоштовно
Формування статистики співробітника	ExpressSJ+mongoDB	Наявна	Доступна безкоштовно
Можливість листування	Websocket+mongoDB	Наявна	Доступна на платній основі з безкоштовною квотою
Таймтрекінг	React	Наявна	Доступна на платній основі з безкоштовною квотою
Створення на редагування задач	React	Наявна	Доступна безкоштовно

Продовження таблиці 4.3

Ідея проекту	Технології реалізації	Наявність технології	Доступність технології
Створення та редагування керівників	React	Наявна	Доступна безкоштовно
Серверна частина	JavaScript, Express.js	Наявна	Доступна безкоштовно
Веб-додаток	JavaScript, React, Redux	Наявна	Доступна безкоштовно

Як, видно, з таблиці вище, для реалізації вебзастосунку не потрібно технологій, що не наявні в даний момент, а також більшість з них є безкоштовними чи умовно безкоштовними, що надає змогу зменшити вартість розробки.

4.3 Розробка маркетингової програми стартапу

У даному розділі описано маркетингову концепцію готового товару, до якого отримають доступ користувачі.

Споживач отримує продукт у вигляді доступу до веб-версії за допомогою свого аккаунту адміністратора. Кожен користувач може створювати лише 1 менеджера та 5 співробітників. Для можливості створення безліч менеджерів та співробітиків, користувач може оформити місячну, піврічну або годову підписку, за допомогою банківської карти, що надасть безлімітний доступ до системи.

У таблиці 4.4 визначено основні переваги використання розроблюваного продукту

Таблиця 4.4 – Основні переваги використання розроблюваного продукту

Потреба	Вигода від використання продукту	Ключові переваги

Система управління та взаємодія співробітників між собою.	Можливість автоматизації роботи з співробітниками, збирання даних для аналізу і поліпшення бізнес-процесів, а також для взаємодії співробітників.	Мінімалістичний та зручний інтерфейс Можливість листування.

Таким чином очевидно, що вебзастосунок має перевагу над потенційними конкурентами, оскільки вони не маюсь можливості листування, але можуть підтримувати її у майбутньому. Також потрібно вибрати системи продажу.

Таблиця 4.5 – Система збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Підписка	Збут та підтримка Тестування Покращення Розробка	нульовий рівень	Пряма (традиційна)

4.4 Аналіз ринкових можливостей запуску стартап-проекту

У даному розділі проаналізовано загрози а також можливості при виході стартап проектк на ринок. Результати аналізу наведено у таблиці 5.6

Таблиц 4.6 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку	Характеристика
1	Кількість головних гравців	2
2	Динаміка ринку	Стагнує
3	Загальний обсяг продаж	Публічна статистика відсутня
4	Наявність обмежень для входу	Відсутні
5	Середня норма рентабельності	Невідома
6	Специфічні вимоги до стандартизації та сертифікації	Немає

4.5 Розробка маркетингової програми стартапу

Таблиця 4.7 – Стратегія маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікації	Ключові позиції для позиціювання	Концепція рекламного звернення
Зменшення вартості наданих послуг та їх покращення	Контекста реклама, веб-сайти	Канал першого рівня	Японська мова, зручний та мінімалістичний інтерфейс

4.6 Розробка ринкової програми стартапу

Таблиця 4.8 – Групи потенційних користувачів продукту

Назва групи	Готовність користуватися проектом	Приблизний попит в межах групи	Інтенсивність конкуренції	Складність входу до сегменту
IT компанії	Готові	Високий	Низька	Середня
Люди, які плануюсь створити свою компанію	Готові	Середній	Низька	Середня
Менеджери, у яких є багато співробітників в підпорядкуванні	Готові	Середній	Низька	Середня

Таблиця 4.9 – Визначення базової стратегії розвитку

Альтернатива розвитку продукту	Стратегія охоплення ринку	Конкурентоспроможні позиції відповідно до альтернативи	Базова стратегія розвитку
Стратегія диференціації	Постійне покращення продуктку на основі відгуків користувачів,	Можливість індивідуалізації продукту під важливих клієнтів	Стратегія спеціалізації

Базовою стратегією було обрано стратегію спеціалізації. На випадок її провалу в подальшому буде використано стратегію диференціації.

У наступній таблиці розглянуто стратегію конкурентної поведінки на ринку.

Таблиця 4.10 – Стратегія конкурентної поведінки на ринку

Чи є проект першепроходцем на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів	Чи буде компанія копіювати основні характеристики товару конкурента, і які	Стратегія конкурентної поведінки
Ні	Нових	Ні, в подальшому можливе копіюванная за умови, що функціх конкурентів будуть потребовані користувачами	Нішева

У таблиці 4.11 представлена стратегія позиціонування продукту на ринку

Таблиця 4.11 – Стратегія позиціонування продукту

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)

Зручний інтерфейс Налаштування під власні потреби Аналіз тексту на японській мові	Заняття ніші	Мінімалістичний інтерфейс Великий набір різних правил аналізу	Гнучкість Швидкість Зручність

4.7 Висновок до розділу

В даному розділі було сформовано та описано ідею розробки стартап проекту. Було проведено досідження можливості реалізації проекту з техінічної точки зору, винайдено маркетинговий план, обрано стратегії розвитку продукту на ринку.

5 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Розгортання програмного забезпечення

Розгортання веб-застосунку складається з наступніх етапів:

- створити базу даних;
- встановити фронтенд-частину;
- встановити бекенд-частину.

5.1.1 Створення бази даних

5.1.1.1 Встановлення MongoDB на Windows

На платформі Windows існує два способи встановлення MongoDB: з пакета установника Microsoft (* .msi) або з архіву. Ми розглянемо обидва варіанти, починаючи з найпростішого - архіву.

Який би спосіб ви не вибрали, результуючий набір двійкових файлів буде однаковим, тому це всього лише питання особистих переваг. Зверніть увагу, що MongoDB не підтримує Windows XP або більш старі версії Windows.

5.1.1.1.1 Встановлення з архіву

Щоб встановити MongoDB з архіву, досить виконати два простих кроки: Завантажити архів дистрибутива:

- Для 32-бітної Windows, будь ласка, скачайте архів з: https://fastdl.mongodb.org/win32/mongodb-win32-i386-2.6.0.zip
- Для Windows 64-bit, будь ласка, завантажте архів з: https://fastdl.mongodb.org/win32/mongodb-win32-x86_64-2008plus-2.6.0.zip

Розпакуйте архів з дистрибутивом.

Після завантаження розпакуйте архів у зручний місце на локальному жорсткому диску.

5.1.1.1.2 Встановлення з пакета MSI

Процедура встановлення MongoDB з готового пакету * .msi також дуже проста:

Завантажити дистрибутив * .msi пакет:

- Для 32-бітної Windows, будь ласка, скачайте пакет * .msi за адресою: https://fastdl.mongodb.org/win32/mongodb-win32-i386-2.6.0.msi
- Для 64-бітної Windows, будь ласка, скачайте пакет * .msi за адресою: https://fastdl.mongodb.org/win32/mongodb-win32-x86_64-2008plus-2.6.0.msi

Запустіть завантажений пакет * .msi.

Дотримуйтесь інструкцій майстра встановлення (при бажанні виберіть зручне місце на локальному жорсткому диску)

Якщо ви вирішили виконати типову установку, місцем призначення за замовчуванням буде C: \ Program Files \ MongoDB 2.6 Standard.

5.1.1.2 Встановлення MongoDB на Linux

Для встановлення MongoDB в Linux відповідні виконавчі файли вже попередньо зібрані і надані в якості дистрибутивних архівів. Давайте пройдемося по процесу установки крок за кроком.

Завантажити архів дистрибутива:

- Для Linux 32-бітний, будь ласка, завантажте архів з: https://fastdl.mongodb.org/linux/mongodb-linux-i686-2.6.0.tgz
- Для Linux 64-bit, будь ласка, завантажте архів з: https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-2.6.0.tgz

Розпакуйте архів з дистрибутивом.

Для Linux 32-бітний дистрибутивний архів: tar xf mongodb-linux-i686-2.6.0.tgz. Повинна бути створена папка: mongodb-linux-i686-2.6.0, що містить всі виконавчі файли.

Для Linux 64-бітний дистрибутивний архів: tar xf mongodb-linux-x86_64-2.6.0.tgz. Повинна бути створена папка: mongodb-linux-x86_64-2.6.0, що містить всі виконавчі файли.

Слід зазначити, що в дистрибутив MongoDB не включені керуючі сценарії для запуску серверного компонента під час запуску системи. Однак MongoDB надає власні офіційні пакети, які розгортають ці сценарії і можуть бути встановлені з використанням різних менеджерів пакетів.

5.1.1.3 Встановлення MongoDB на MacOS

Встановлення на MacOS дуже схожа на версію Linux за винятком того факту, що надається тільки 64-бітний дистрибутив. Давайте докладніше розглянемо етапи встановлення.

Завантажте архів дистрибутива: https://fastdl.mongodb.org/osx/mongodb-osx-x86_64-2.6.0.tgz .

Розпакуйте архів з дистрибутивом.

Після завантаження, будь ласка, запустіть команду:

tar xf mongodb-osx-x86_64-2.6.0.tgz

Повинна бути створена папка mongodb-osx-x86_64-2.6.0, що містить всі виконавчі файли.

5.1.1.4 Запуск MongoDB

Для запуску бази данних, потрібно лише відкрити консоль та ввести в неї наступні команди:

- brew install mongodb-community;
- brew services run mongodb-community.

5.1.2 Встановлення фронтенд частини

Для встановлення фронтенд частини потрібно відкрити консоль саме в папці фронтенд частини та ввести наступні команди:

- npm install;
- npm start;

Після цього у вас повинен відкритися браузер з запущеним вебзастосунком.

5.1.3 Встановлення бекенд частини

Для встановлення бекенд частини потрібно відкрити консоль саме в папці бекенд частини та ввести наступні команди:

- npm install;
- npm start;

5.2 Інструкція користувача для адміністратора

Для початку роботи з вебзастосунком адміністратору потрібно пройти процесс авторизації. Для цього на головний сторінці системи потрібно натиснути кнопку Admin (Рисунок 5.1).

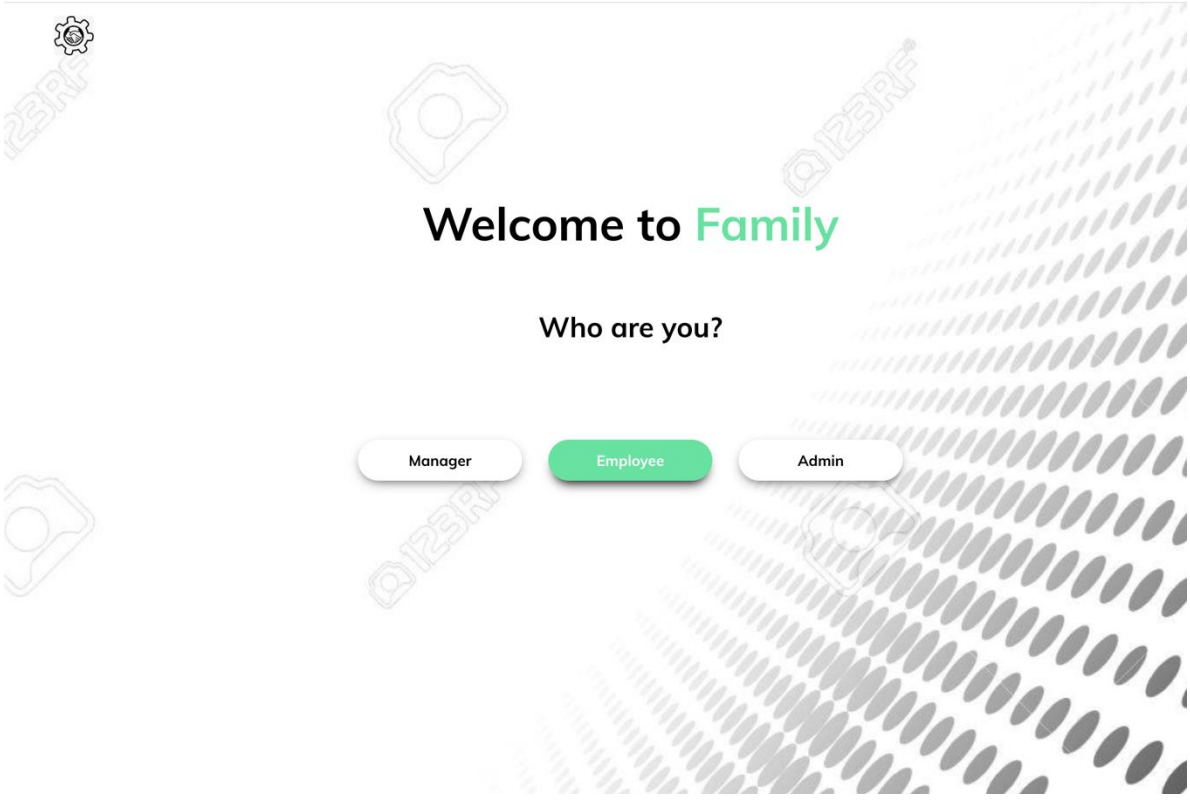

Рисунок 5.1 – Головна сторінка вебзастосунку для адміністратора

Після цього на сторінкі авторизації для адміністратора, ввести дані та натиснути кнопку Login (Рисунок 5.2).

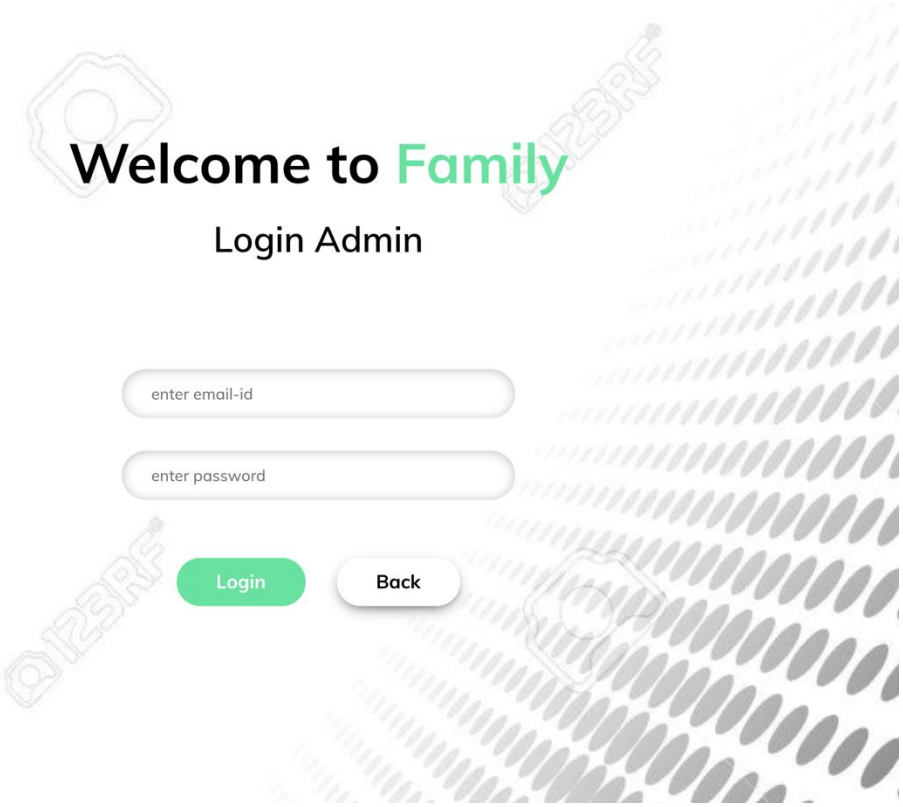

Рисунок 5.2 – Сторінка авторизації адміністратора

Адміністратор потрапить на головну сторінку вебзастосунку. (Рисунок 5.3)

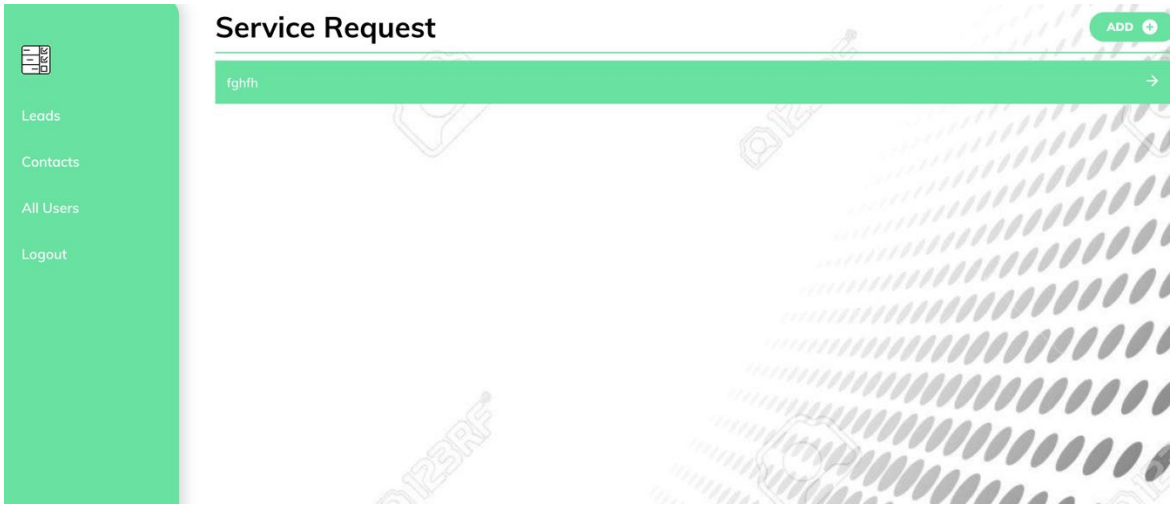

Рисунок 5.3 - Головна сторінка системи для адміністратора після успішного проходження авторизації

На лівій частині головної сторінки знаходиться меню. На правій частині знаходиться інформація з вибраного меню. Після авторизації, адміністратор потрапляє на сторінку меню «Задачі». На цій сторінці адміністратор має

можливість переглядати усі створені задачі, також він може переглянути задачу більш детально натиснувши на стрілку справа на конктерній задачі. Після цього адміністратор потрапить на сторінку детальной інформації про задачу (Рисунок 5.4).

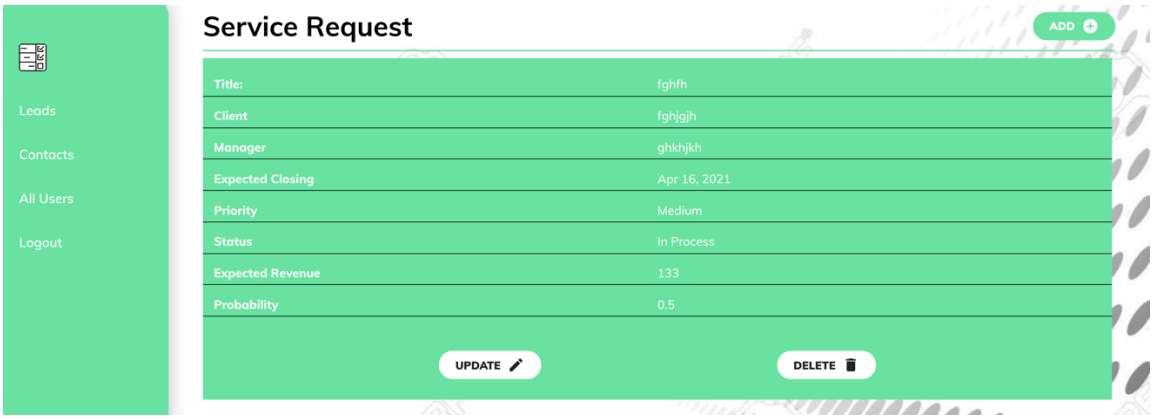

Рисунок 5.4 – Сторінка детальної інформації про задачу

При натисканні на кнопку Delete адміністратор вилалить вибрану задачу, та повернеться на сторінку з усіма задачами. При натисканні на кнопку Update адміністратор попаде на сторінку з можливістю редагувати інформацію про задачу (Рисунок 5.5).

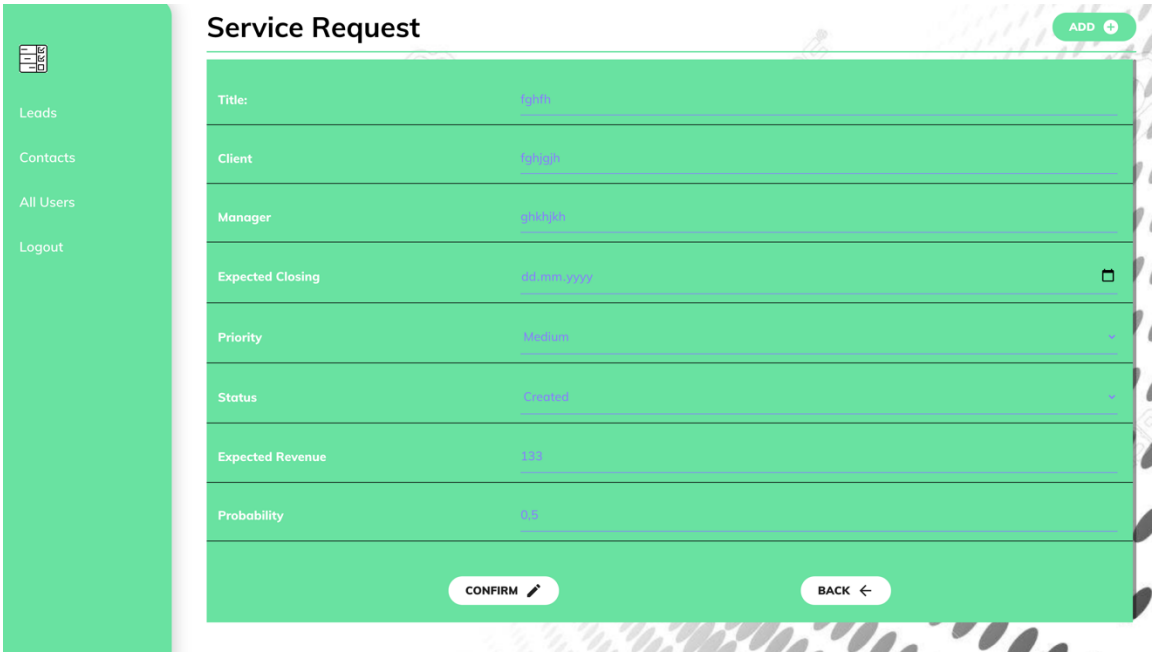

Рисунок 5.5 – Сторінка редагування інформації про задачу

Далі, щоб підтвердити редагування, адміністратор повинен натиснути кнопку Confirm, або щоб відмінити редагування- кнопку Back, після чого він повернется на сторінку с задачами.

Також адміністратор має можливість створювати нові задачі, натиснувши на кнопку Add (Рисунок 5.4). Після цього він перейде на сторінку створення нової задачі (Рисунок 5.5)

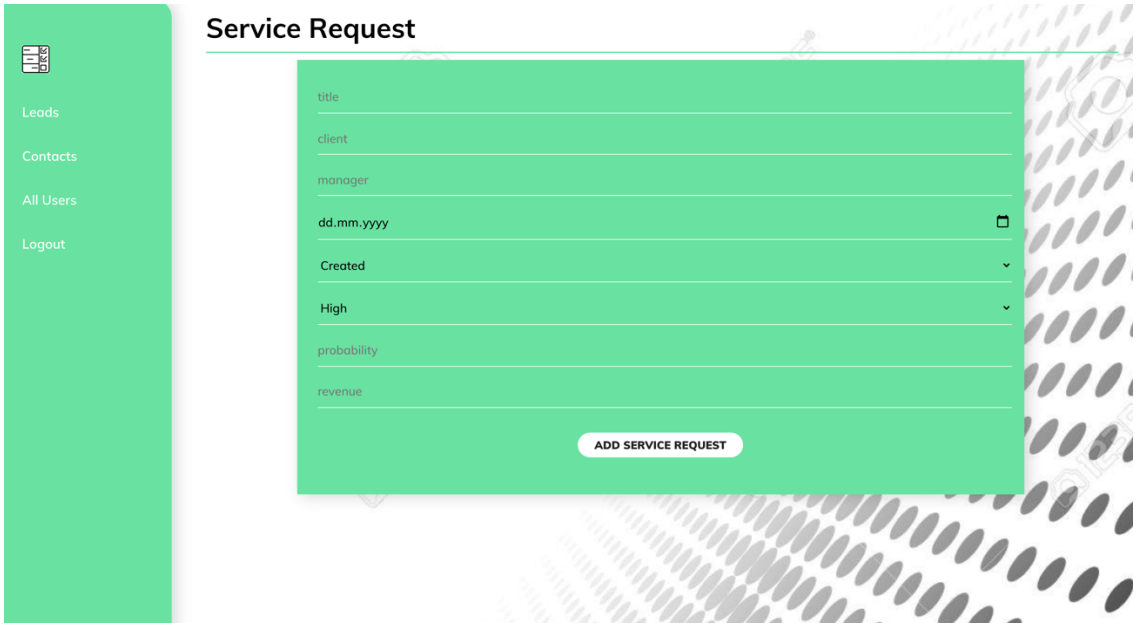

Рисунок 5.5 – Сторінка створення нової задачі

Після того, як вся інформація буде заповнена, адміністратор повинен натиснути кнопку Add Service Request, тим самим він створить нову задачу та повернеться на сторінку с задачами.

Наступна сторінка меню – це сторінка керівників (Рисунок 5.6)

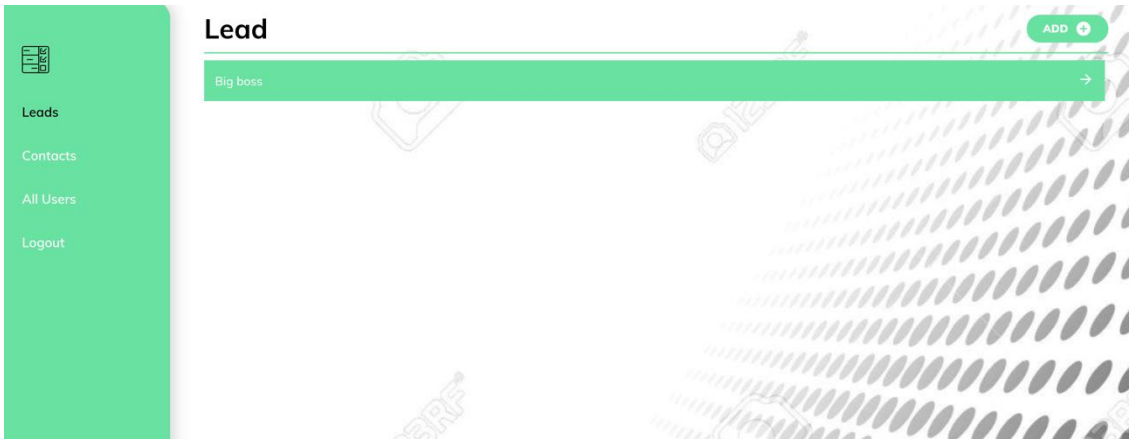

Рисунок 5.6 – Сторінка Керівників

На цій сторінці адміністратор має можливість переглядати всіх керівників, щоб отримати більш детальну інформацію про керівника, потрібно натиснути на стрілку праворуч від ім’я (Рисунок 5.6) та адміністратор перейде на сторінку детальної інформації про керівника (Рисунок 5.7).

Рисунок 5.7 – Сторінка детальної інформації про керівника

На цій сторінці адміністратор має можливість переглядати детальну інформацію про керівника, також може його видалити, натиснувши на кнопку Delete, або змінити інформацію про керівника, натиснувши на кнопку Update.

Натиснувши на кнопку Update, адміністратор перейде на сторінку редагування інформації про керівника (Рисунок 5.8).

Рисунок 5.8 - Сторінка редагування інформації про керівника

На цій сторінці – адміністратор має можливість змінювати інформацію про керівника. Далі, щоб підтвердити редагування, адміністратор повинен натиснути кнопку Confirm, або щоб відмінити редагування - кнопку Back, після чого він повернется на сторінку с керівниками.

Також адміністратор має можливість додавати нових керівників, натиснувши на кнопку Add (Рисунок 5.8). Після цього він перейде на сторінку додавання нового керівника (Рисунок 5.9)

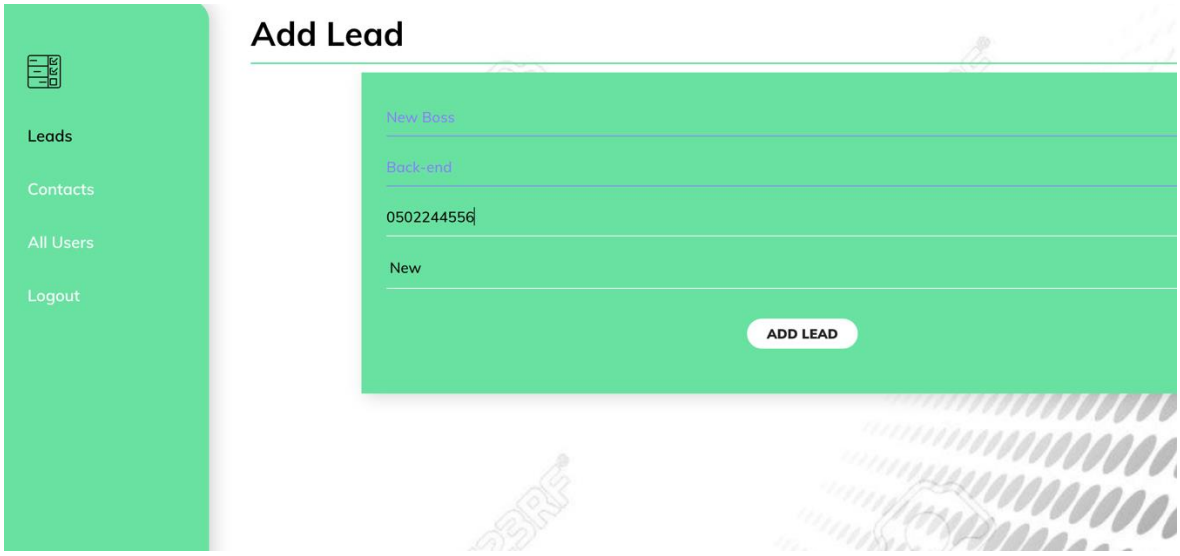

Рисунок 5.9 - Сторінка додавання нового керівника

Після того, як вся інформація буде заповнена, адміністратор повинен натиснути кнопку Add Lead, тим самим він підтвердить додавання нового керівника та повернеться на сторінку с керівниками.

Наступна сторінка меню – це сторінка з клієнтами компанії(Рисунок 5.10)

Рисунок 5.10 – Сторінка з клієнтами компанії

На цій сторінці адміністратор має можливість переглядати всіх клієнтів, щоб отримати більш детальну інформацію про клієнта, потрібно натиснути на стрілку праворуч від ім’я (Рисунок 5.10) та адміністратор перейде на сторінку детальної інформації про клієнта (Рисунок 5.11).

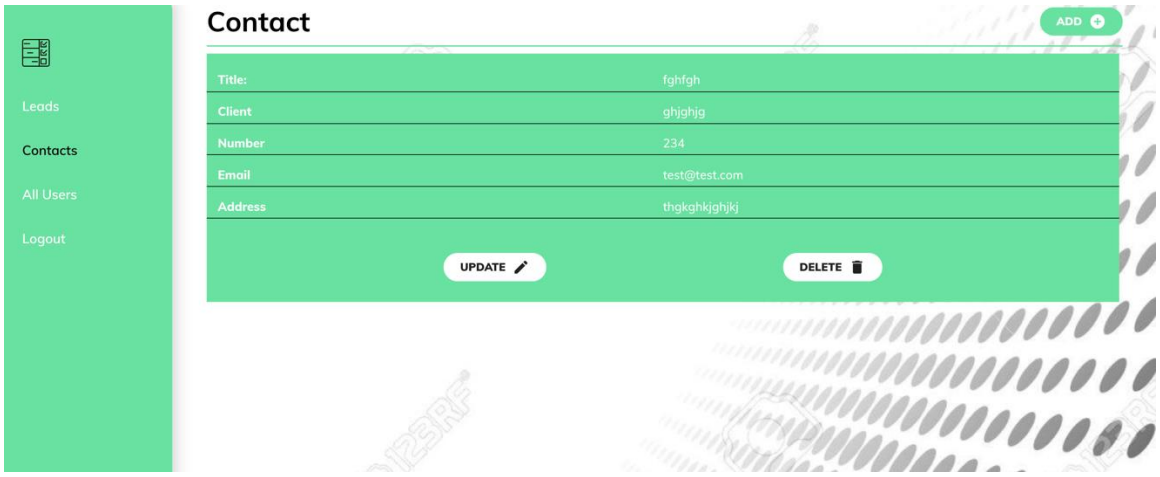

Рисунок 5.11 – Сторінка детальної інформації про клієнта для адміністратора

На цій сторінці адміністратор має можливість переглядати детальну інформацію про клієнта, також може його видалити, натиснувши на кнопку Delete, або змінити інформацію про клієнта, натиснувши на кнопку Update.

Натиснувши на кнопку Update, адміністратор перейде на сторінку редагування інформації про клієнта (Рисунок 5.12).

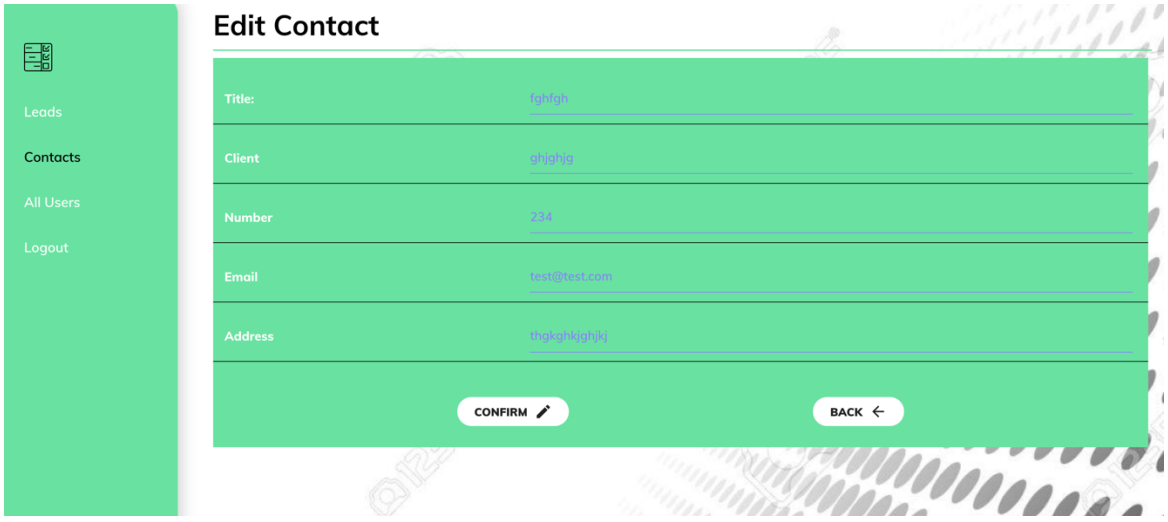

Рисунок 5.12 - Сторінка редагування інформації про клієнта для адміністратора

На цій сторінці – адміністратор має можливість змінювати інформацію про клієнта. Далі, щоб підтвердити редагування, адміністратор повинен натиснути кнопку Confirm, або щоб відмінити редагування - кнопку Back, після чого він повернеться на сторінку з клієнтами.

Також адміністратор має можливість додавати нових клієнтів, натиснувши на кнопку Add (Рисунок 5.12). Після цього він перейде на сторінку додавання нового клієнта (Рисунок 5.13)

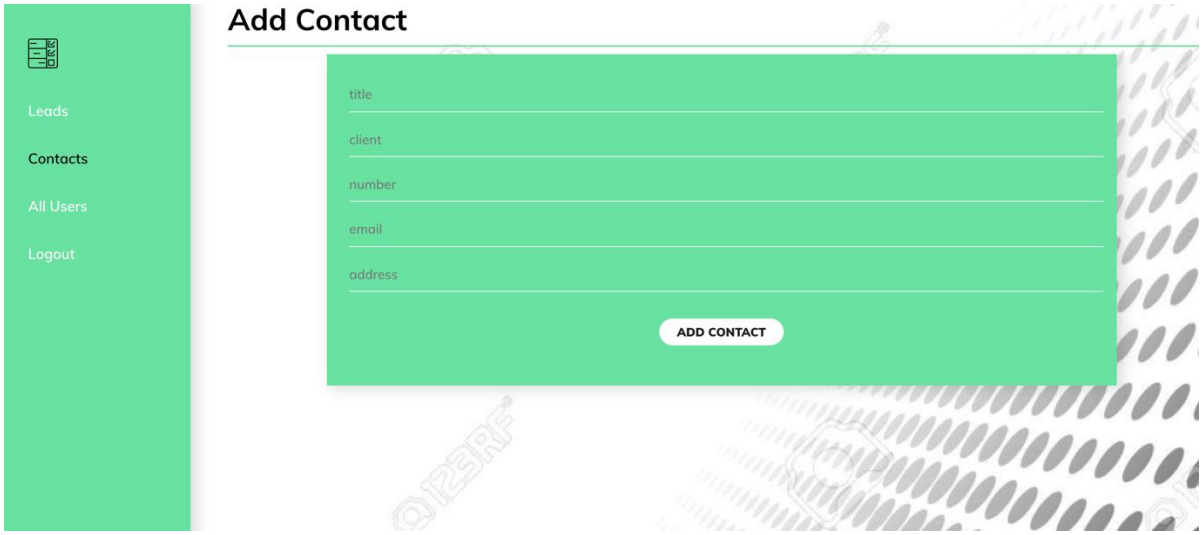

Рисунок 5.13 - Сторінка додавання нового клієнта для адміністратора

Після того, як вся інформація буде заповнена, адміністратор повинен натиснути кнопку Add Contact, тим самим він підтвердить додавання нового клієнта та повернеться на сторінку с клієнтами.

Наступна сторінка меню – це сторінка усіх користувачів (Рисунок 5.14)

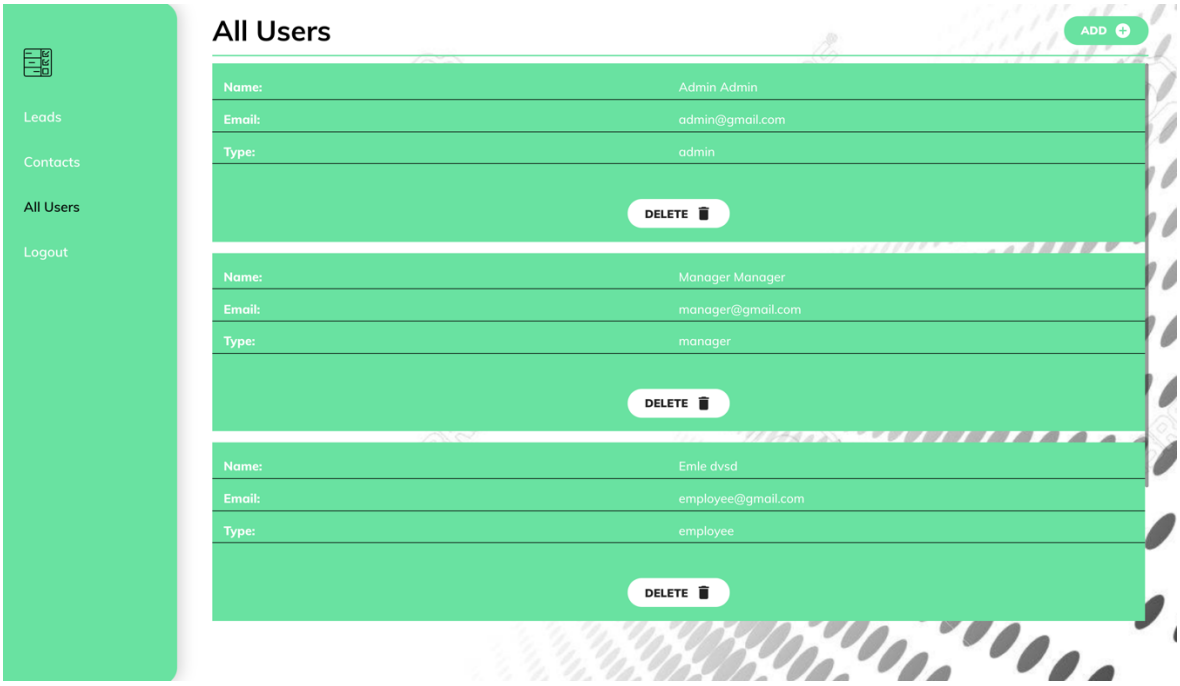

Рисунок 5.14 - Сторінка усіх користувачів

На цій сторінці – адміністратор має можливість переглянути список усіх користувачів, які мають доступ до вебзастосунку. Також, він може видаляти користувачів натиснувши кнопку delete, або додавати нових, натиснувши кнопку add. Після цього він перейде на сторінку додавання нового користувача (Рисунок 5.15)

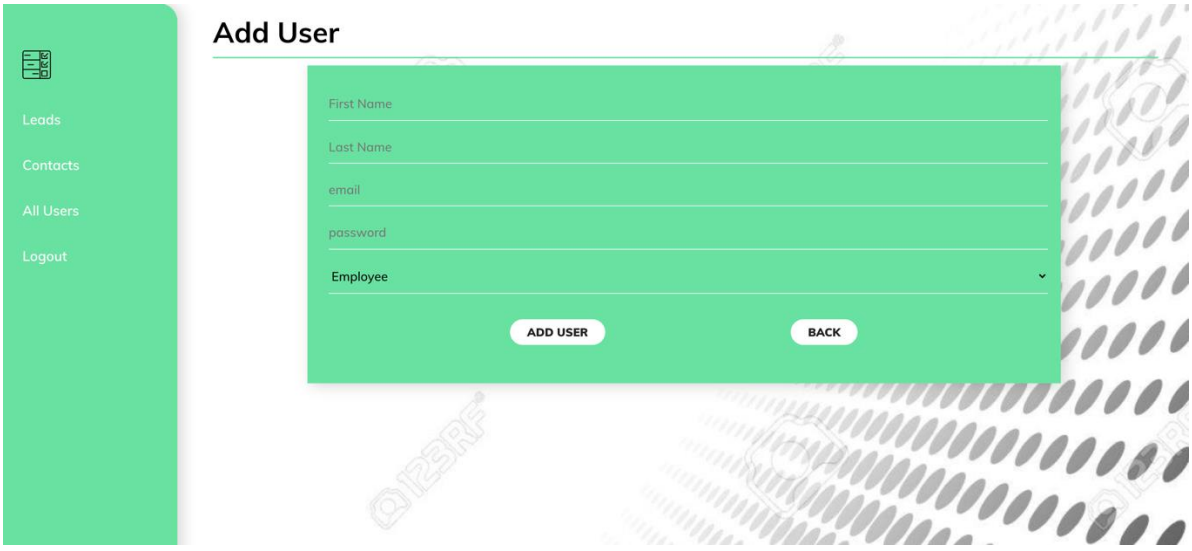

Рисунок 5.15 - Сторінка додавання нового користувача

Після того, як вся інформація буде заповнена, адміністратор повинен натиснути кнопку Add User, тим самим він підтвердить додавання нового користувача та повернеться на сторінку с користувачами.

Також адміністратор має можливість виходу с аккаунту, натиснувши кнопку Logout (Рисунок 5.16).

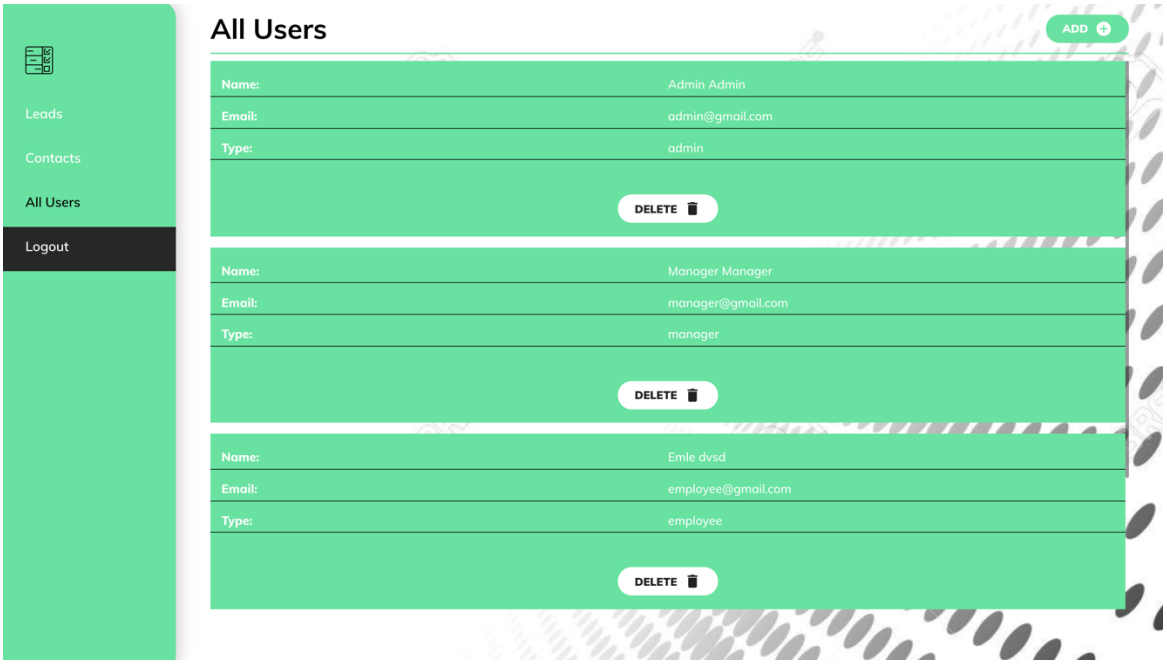

Рисунок 5.16 – Кнопка виходу з аккаунту

5.3 Інструкція користувача для співробітника

Для початку роботи з вебзастосунком співробітнику потрібно пройти процесс авторизації. Для цього на головний сторінці системи потрібно натиснути кнопку Employee (Рисунок 5.17).

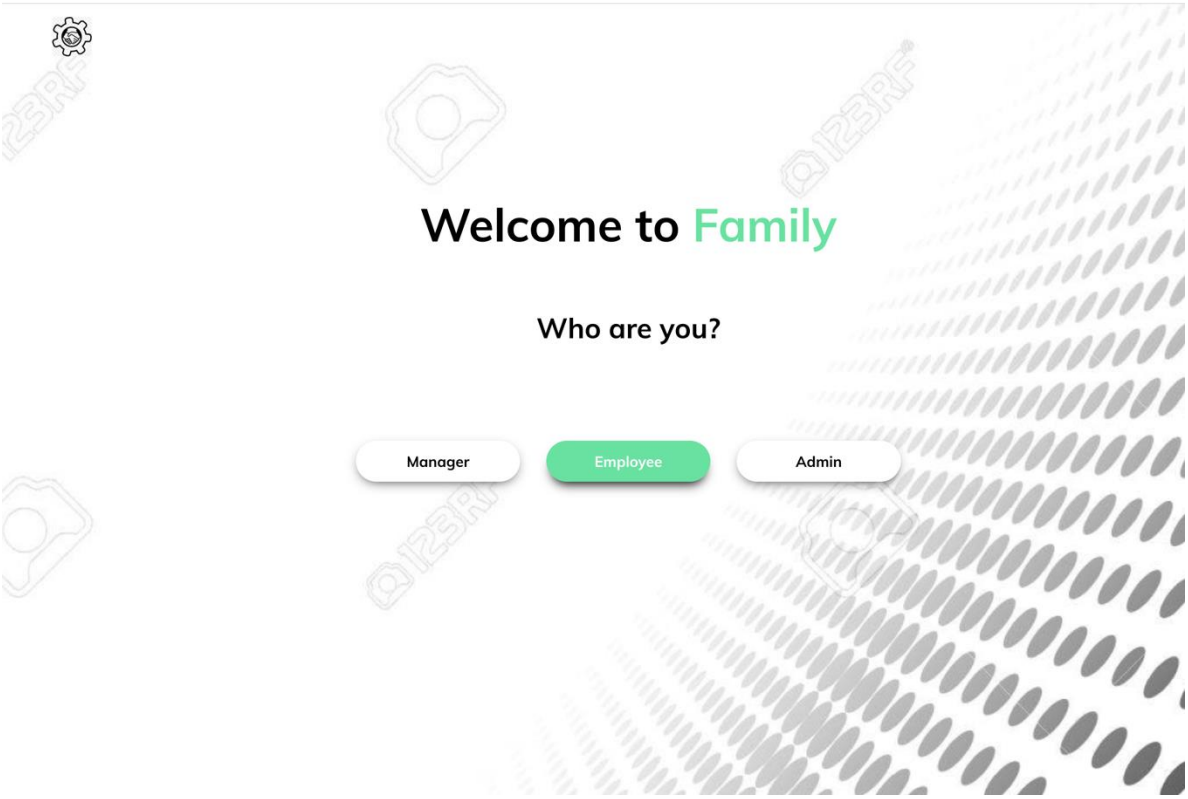

Рисунок 5.17 – Головна сторінка вебзастосунку для співробітника

Після цього на сторінкі авторизації для співробітника, ввести дані та натиснути кнопку Login (Рисунок 5.18).

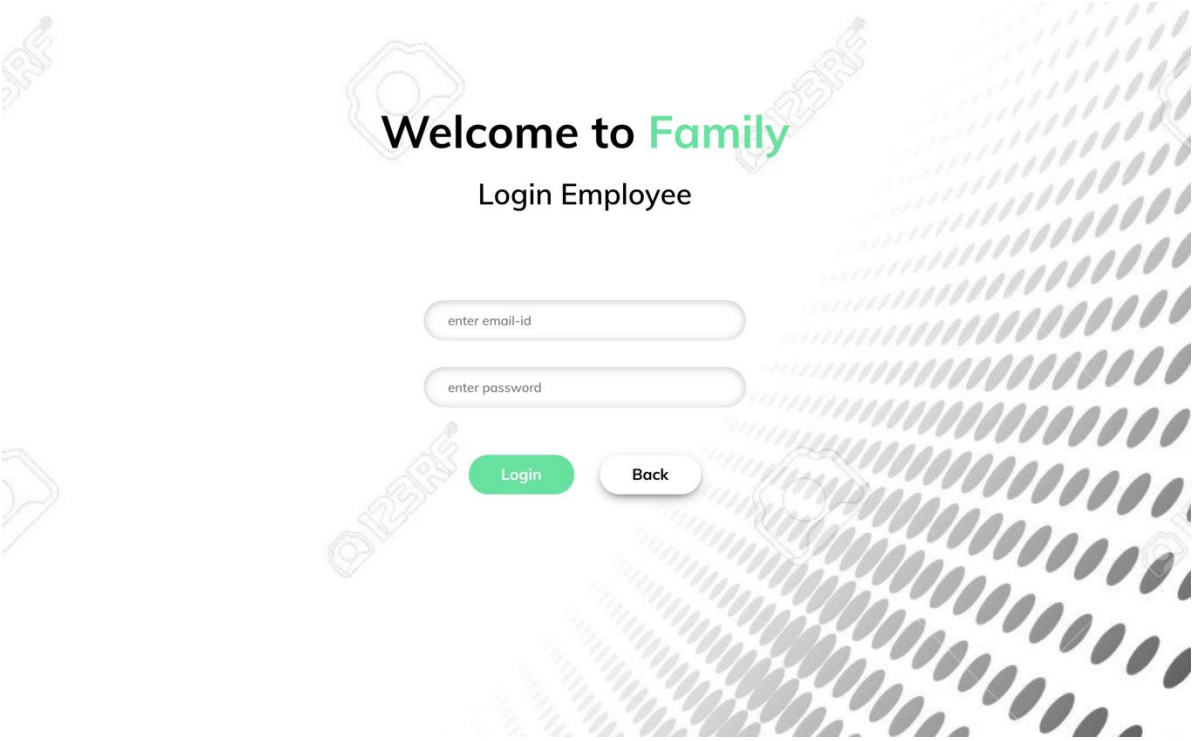

Рисунок 5.18 – Сторінка авторизації співробітника

Співробітник потрапить на головну сторінку вебзастосунку. (Рисунок 5.19)

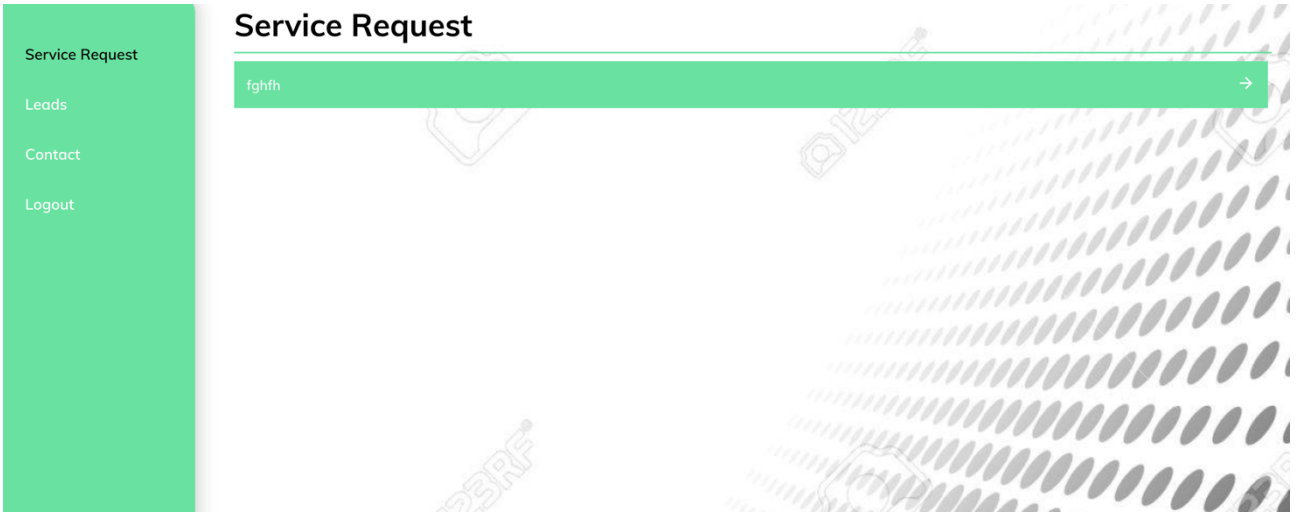

Рисунок 5.19 – Головна сторінка системи для співробітника після успішного проходження авторизації

На лівій частині головної сторінки знаходиться меню. На правій частині знаходиться інформація з вибраного меню. Після авторизації, співробітник потрапляє на сторінку меню «Задачі». На цій сторінці співробітник має можливість переглядати усі створені задачі, також він може переглянути задачу більш детально натиснувши на стрілку справа на конктерній задачі. Після цього співробітник потрапить на сторінку детальной інформації про задачу (Рисунок 5.20).

Рисунок 5.20 – Сторінка детальної інформації про задачу для співробітника

Наступна сторінка меню – це сторінка керівників (Рисунок 5.21)

Рисунок 5.21 – Сторінка Керівників для співробітника

На цій сторінці співробітник має можливість переглядати всіх керівників, щоб отримати більш детальну інформацію про керівника, потрібно натиснути на стрілку праворуч від ім’я (Рисунок 5.21) та співробітник перейде на сторінку детальної інформації про керівника (Рисунок 5.22).

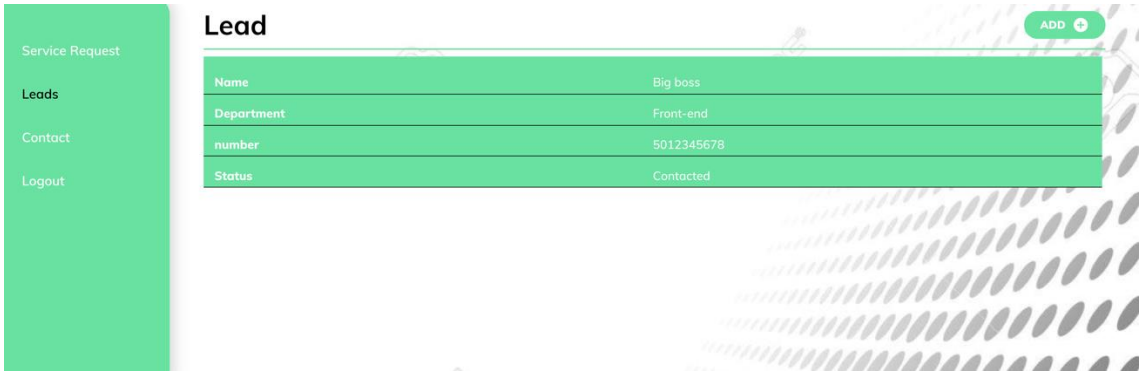

Рисунок 5.22 – Сторінка детальної інформації про керівника для співробітника.

На цій сторінці співробітник має можливість переглядати детальну інформацію про керівника.

Наступна сторінка меню – це сторінка з клієнтами компанії(Рисунок 5.23)

Рисунок 5.23 – Сторінка з клієнтами компанії для співробітника

На цій сторінці співробітник має можливість переглядати всіх клієнтів, щоб отримати більш детальну інформацію про клієнта, потрібно натиснути на стрілку праворуч від ім’я (Рисунок 5.23) та співробітник перейде на сторінку детальної інформації про клієнта (Рисунок 5.24).

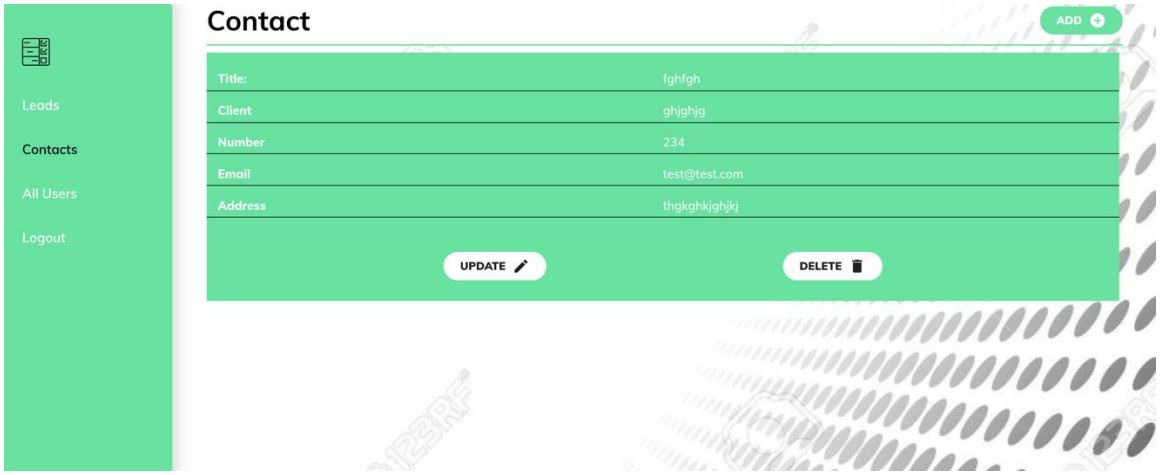

Рисунок 5.24 – Сторінка детальної інформації про клієнта для адміністратора

На цій сторінці співробітник має можливість переглядати детальну інформацію про клієнта.

Також співробітник має можливість виходу с аккаунту, натиснувши кнопку Logout (Рисунок 5.25).

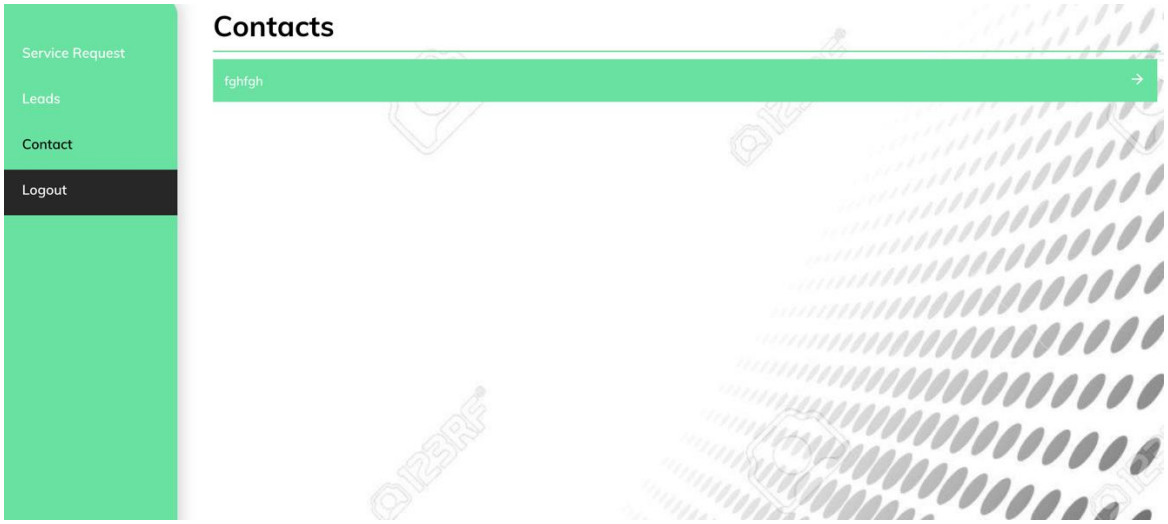

Рисунок 5.25 – Кнопка виходу з аккаунту для співробітника

5.4 Інструкція користувача для менеджера

Для початку роботи з вебзастосунком менеджеру потрібно пройти процесс авторизації. Для цього на головний сторінці системи потрібно натиснути кнопку Manager (Рисунок 5.26).

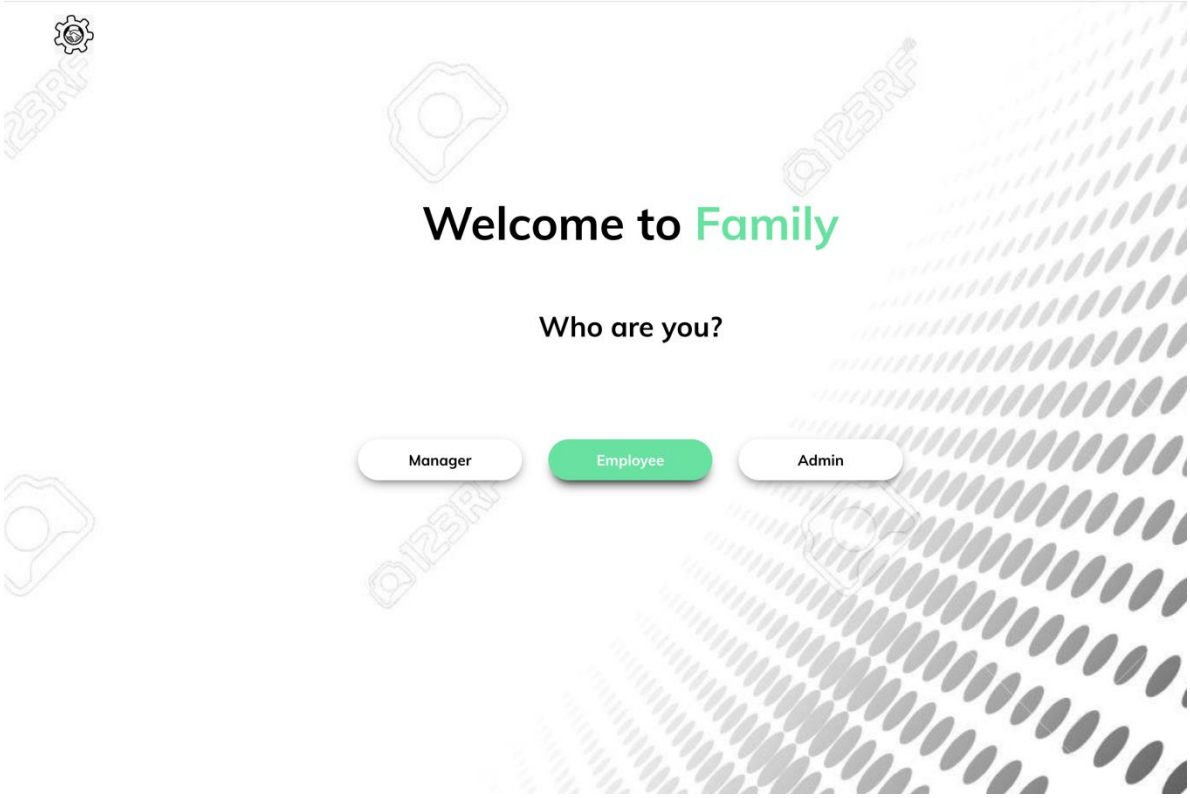

Рисунок 5.26 – Головна сторінка вебзастосунку для менеджера

Після цього на сторінкі авторизації для менеджера, ввести дані та натиснути кнопку Login (Рисунок 5.2).

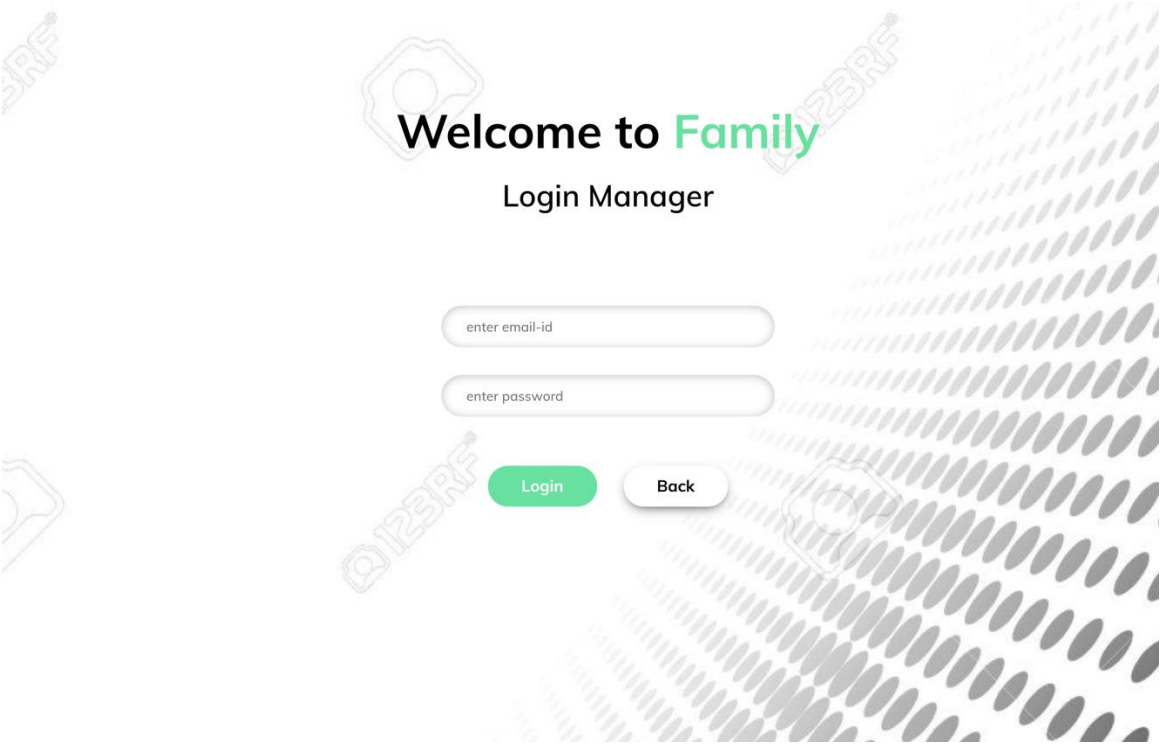

Рисунок 5.2 – Сторінка авторизації менеджера

Менеджер потрапить на головну сторінку вебзастосунку. (Рисунок 5.3)

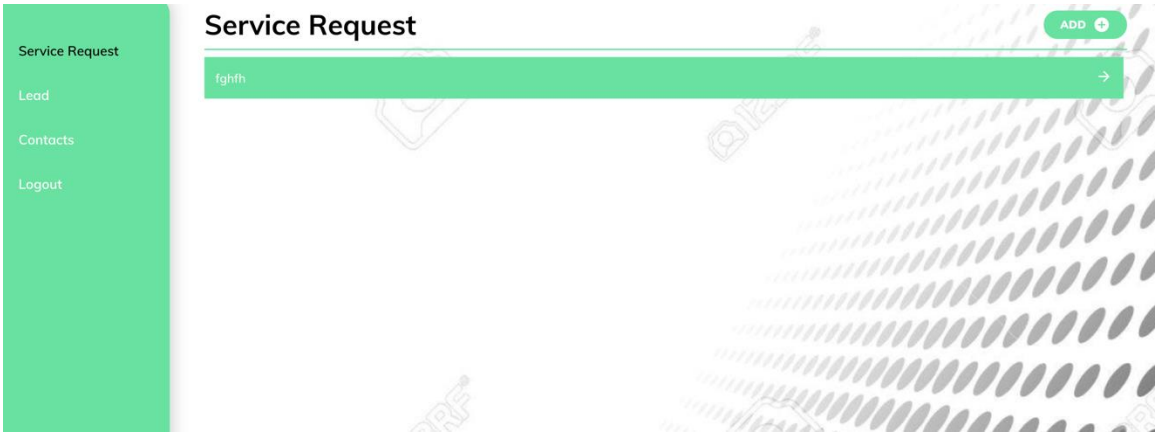

Рисунок 5.3 - Головна сторінка системи для менеджера після успішного проходження авторизації

На лівій частині головної сторінки знаходиться меню. На правій частині знаходиться інформація з вибраного меню. Після авторизації, менеджер потрапляє на сторінку меню «Задачі». На цій сторінці менеджер має можливість переглядати усі створені задачі, також він може переглянути задачу більш

детально натиснувши на стрілку справа на конктерній задачі. Після цього менеджер потрапить на сторінку детальной інформації про задачу (Рисунок 5.27).

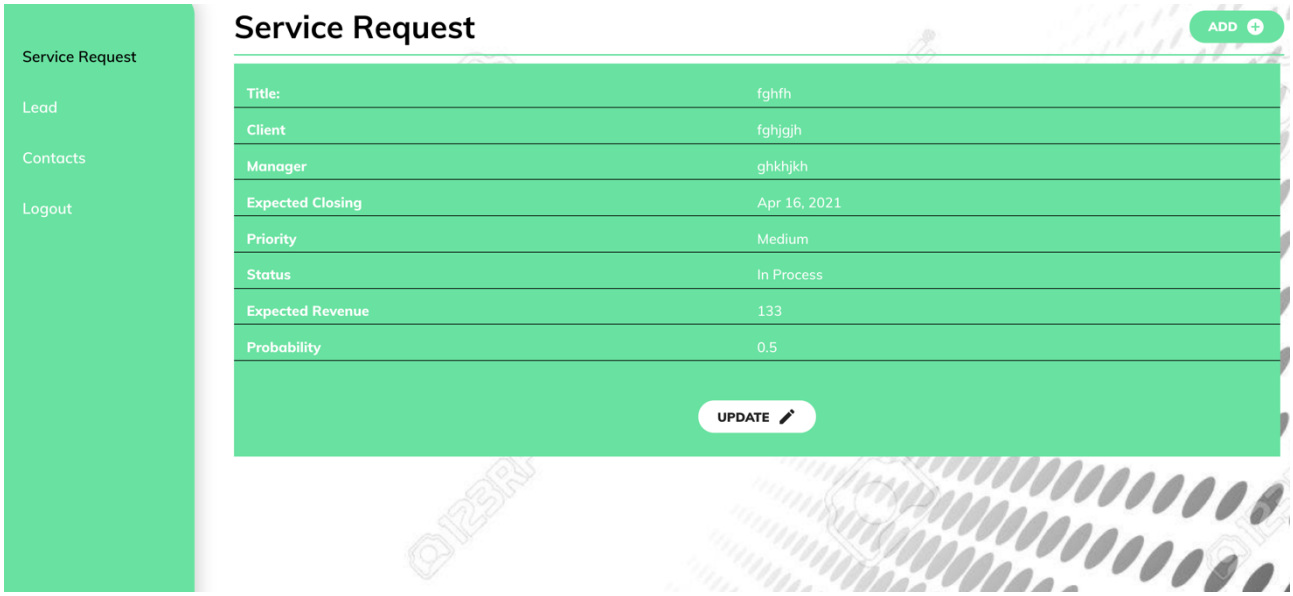

Рисунок 5.27 – Сторінка детальної інформації про задачу для менеджера

При натисканні на кнопку Delete менеджер видалить вибрану задачу, та повернеться на сторінку з усіма задачами. При натисканні на кнопку Update менеджер попаде на сторінку з можливістю редагувати інформацію про задачу (Рисунок 5.28).

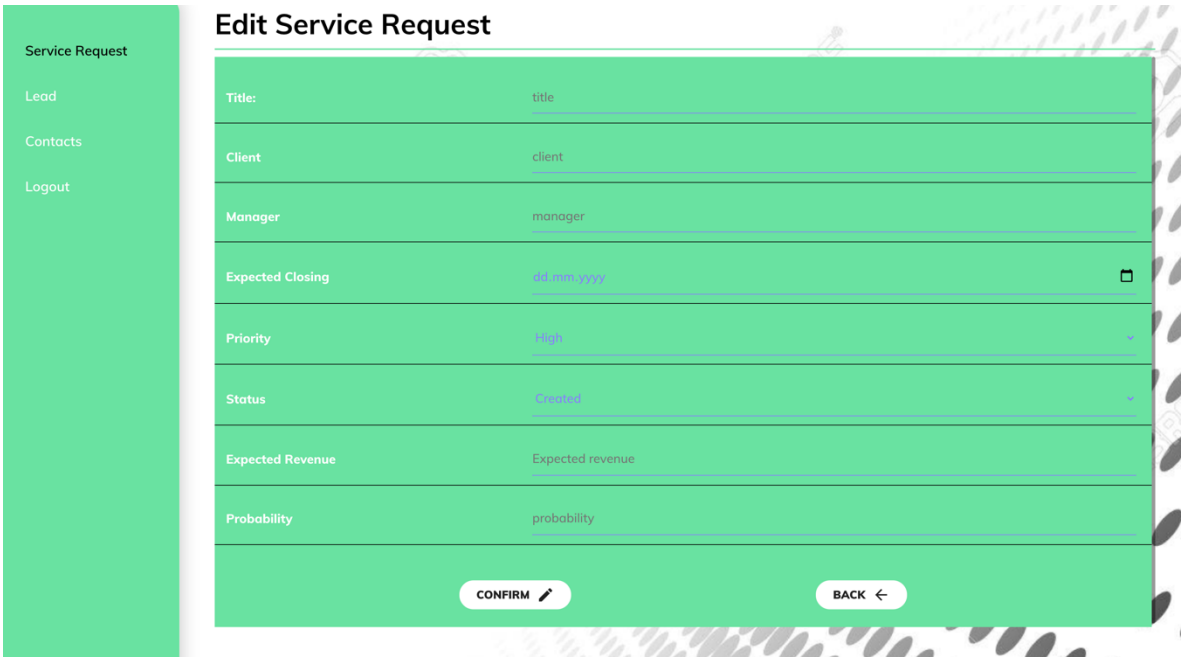

Рисунок 5.28 – Сторінка редагування інформації про задачу для менеджера

Далі, щоб підтвердити редагування, менеджер повинен натиснути кнопку Confirm, або щоб відмінити редагування- кнопку Back, після чого він повернется на сторінку с задачами.

Також менеджер має можливість створювати нові задачи, натиснувши на кнопку Add (Рисунок 5.28). Після цього він перейде на сторінку створення нової задачі (Рисунок 5.29)

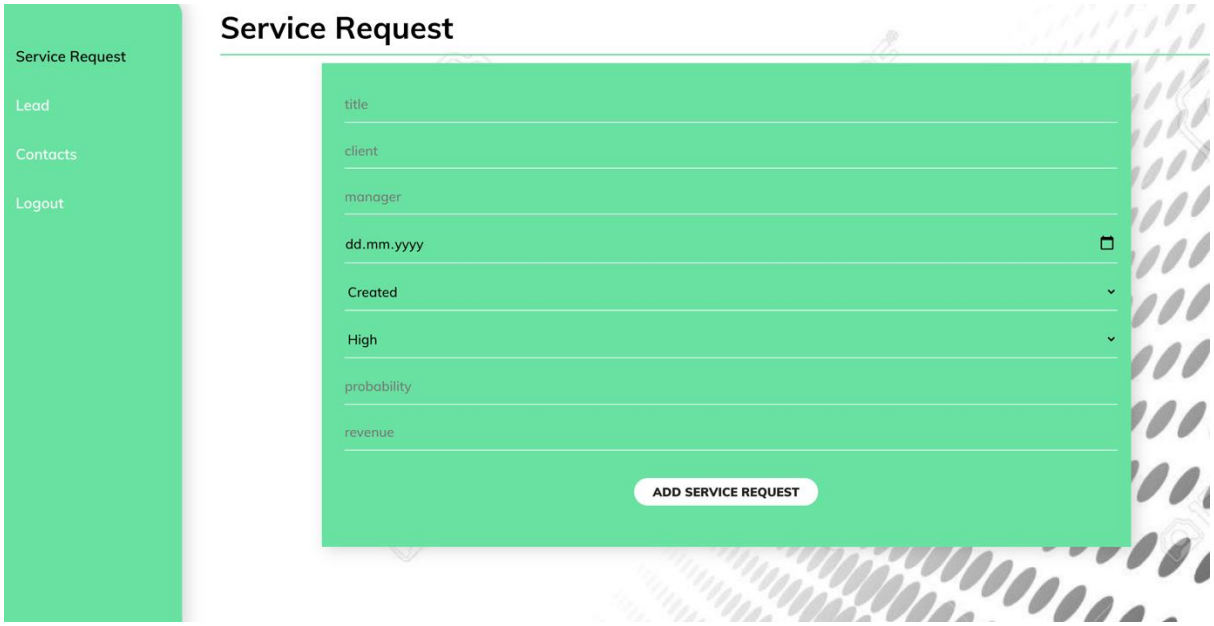

Рисунок 5.29 – Сторінка створення нової задачі

Після того, як вся інформація буде заповнена, менеджер повинен натиснути кнопку Add Service Request, тим самим він створить нову задачу та повернеться на сторінку с задачами.

Наступна сторінка меню – це сторінка керівників (Рисунок 5.30)

Рисунок 5.30 – Сторінка Керівників для менеджера

На цій сторінці менеджер має можливість переглядати всіх керівників, щоб отримати більш детальну інформацію про керівника, потрібно натиснути на стрілку праворуч від ім’я (Рисунок 5.30) та менеджер перейде на сторінку детальної інформації про керівника (Рисунок 5.31).

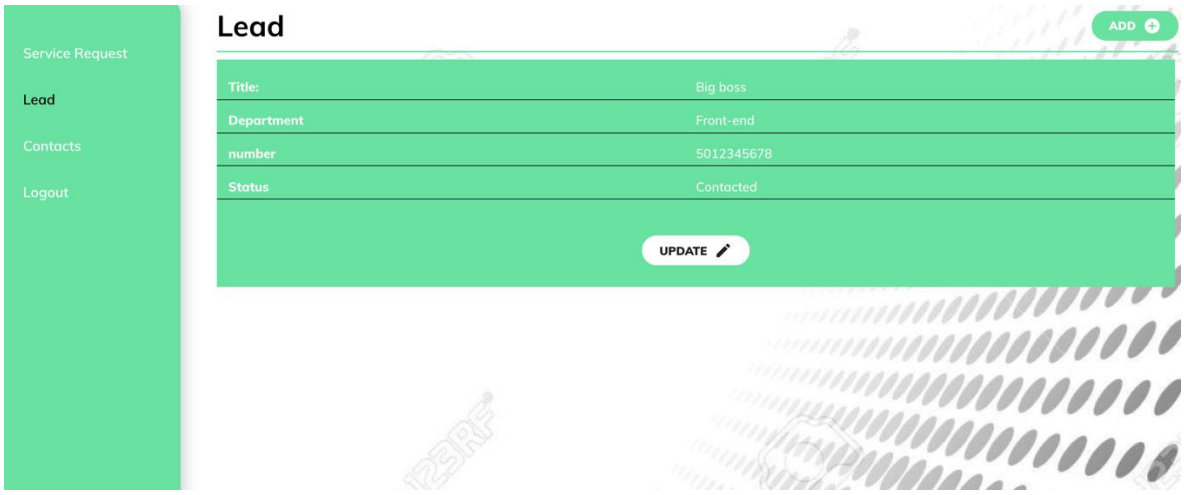

Рисунок 5.31 – Сторінка детальної інформації про керівника для менеджера

На цій сторінці менеджер має можливість переглядати детальну інформацію про керівника, також може його видалити, натиснувши на кнопку Delete, або змінити інформацію про керівника, натиснувши на кнопку Update.

Натиснувши на кнопку Update, менеджер перейде на сторінку редагування інформації про керівника (Рисунок 5.32).

Рисунок 5.32 - Сторінка редагування інформації про керівника для менеджера

На цій сторінці – менеджер має можливість змінювати інформацію про керівника. Далі, щоб підтвердити редагування, менеджер повинен натиснути кнопку Confirm, або щоб відмінити редагування - кнопку Back, після чого він повернется на сторінку с керівниками.

Також менеджер має можливість додавати нових керівників, натиснувши на кнопку Add (Рисунок 5.32). Після цього він перейде на сторінку додавання нового керівника (Рисунок 5.33)

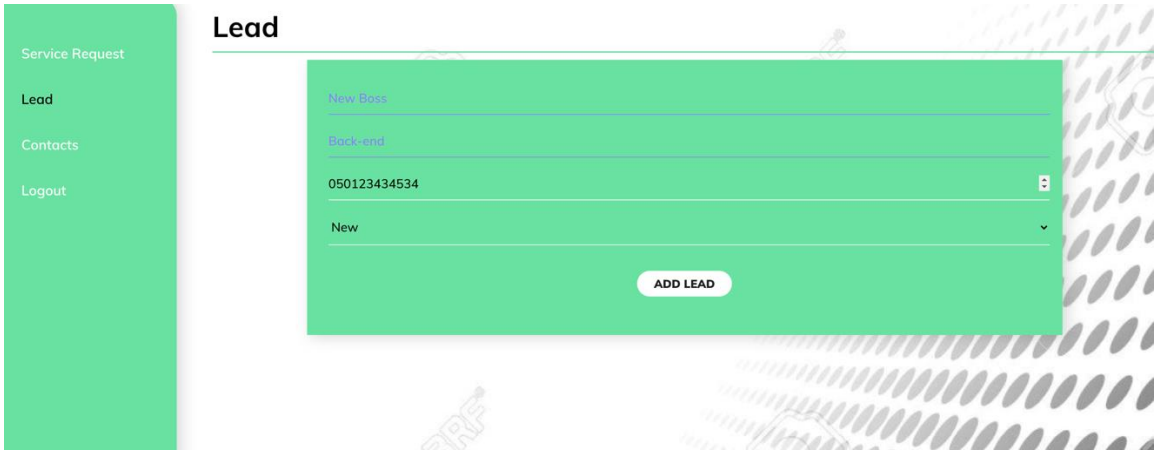

Рисунок 5.33 - Сторінка додавання нового керівника для менеджера

Після того, як вся інформація буде заповнена, менеджер повинен натиснути кнопку Add Lead, тим самим він підтвердить додавання нового керівника та повернеться на сторінку с керівниками.

Наступна сторінка меню – це сторінка з клієнтами компанії(Рисунок 5.34)

Рисунок 5.34 – Сторінка з клієнтами компанії для менеджера

На цій сторінці менеджер має можливість переглядати всіх клієнтів, щоб отримати більш детальну інформацію про клієнта, потрібно натиснути на стрілку

праворуч від ім’я (Рисунок 5.34) та менеджер перейде на сторінку детальної інформації про клієнта (Рисунок 5.35).

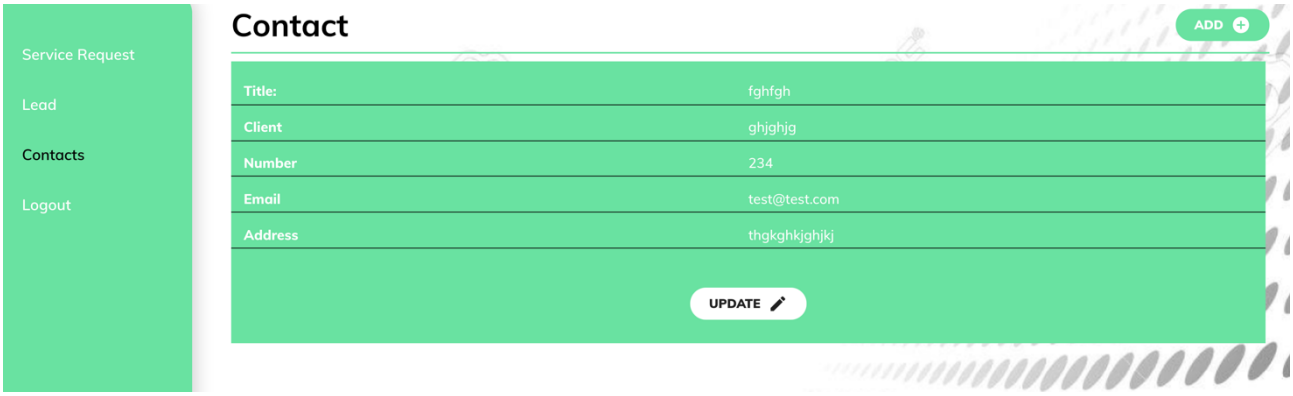

Рисунок 5.35 – Сторінка детальної інформації про клієнта для менеджера

На цій сторінці менеджер має можливість переглядати детальну інформацію про клієнта, також може змінити інформацію про клієнта, натиснувши на кнопку Update.

Натиснувши на кнопку Update, менеджер перейде на сторінку редагування інформації про клієнта (Рисунок 5.36).

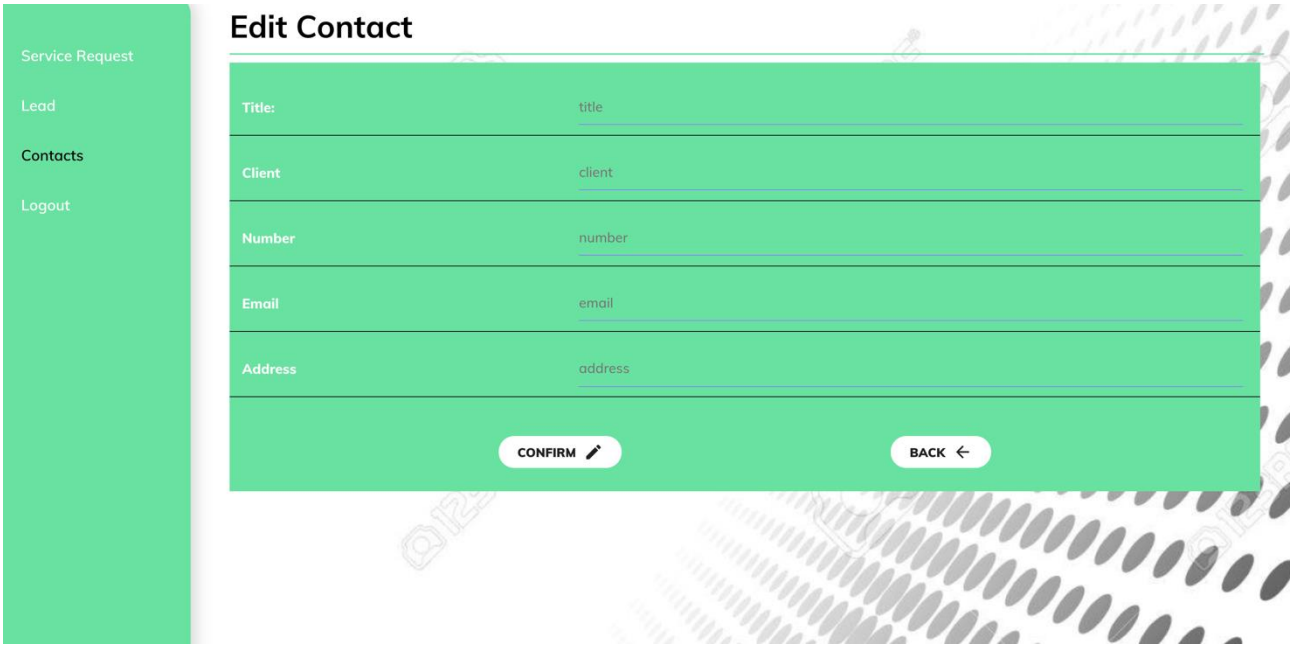

Рисунок 5.36 - Сторінка редагування інформації про клієнта для менеджера

На цій сторінці – менеджер має можливість змінювати інформацію про клієнта. Далі, щоб підтвердити редагування, менеджер повинен натиснути

кнопку Confirm, або щоб відмінити редагування - кнопку Back, після чого він повернеться на сторінку з клієнтами.

Також менеджер має можливість додавати нових клієнтів, натиснувши на кнопку Add (Рисунок 5.36). Після цього він перейде на сторінку додавання нового клієнта (Рисунок 5. 37)

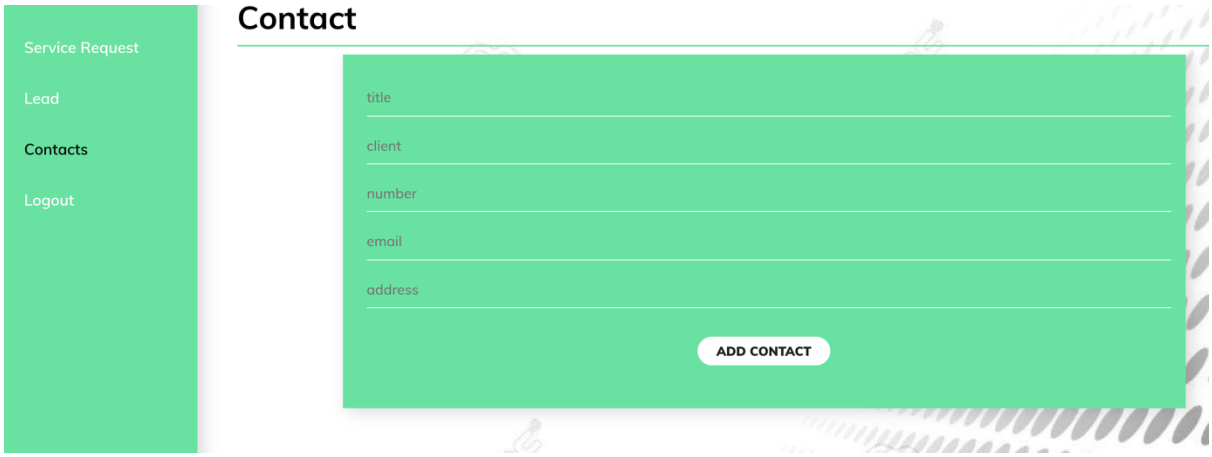

Рисунок 5.37 - Сторінка додавання нового клієнта для менеджера

Після того, як вся інформація буде заповнена, менеджер повинен натиснути кнопку Add Contact, тим самим він підтвердить додавання нового клієнта та повернеться на сторінку с клієнтами.

Також менеджер має можливість виходу с аккаунту, натиснувши кнопку Logout (Рисунок 5.38).

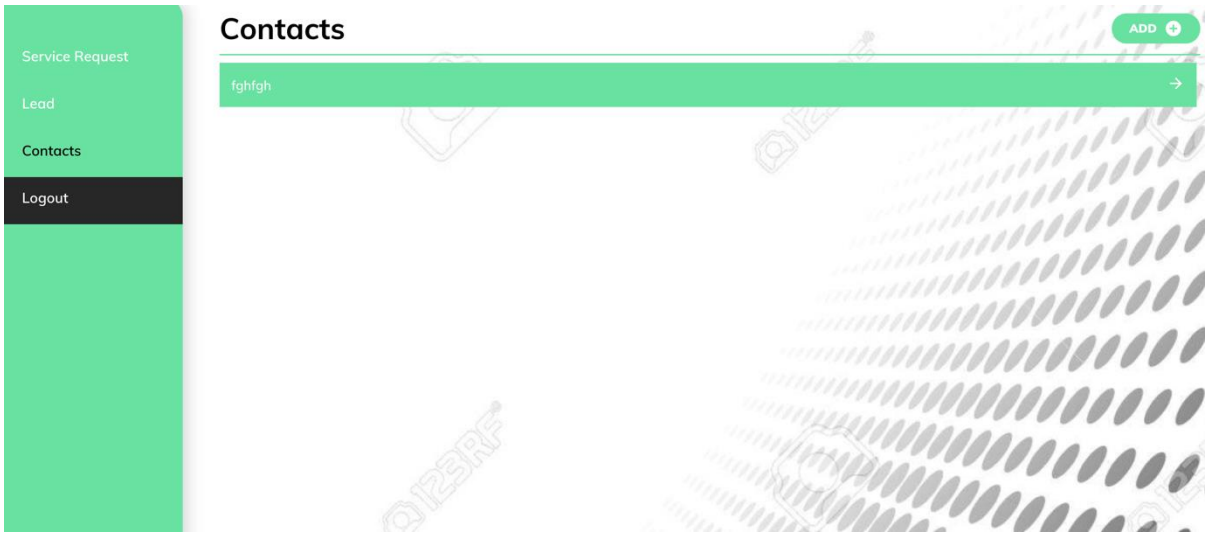

Рисунок 5.38 – Кнопка виходу з аккаунту для менеджера

5.5 Висновки по розділу

В даному розділі було описано процесс встановлення вебзастосунку, а також розроблена інструкція для користувача – адміністратора, користувача – співробітника та користувача – менеджера.

ВИСНОВКИ

У ході написання магістерської роботи на тему «Веб-застосунок для онлайн управління та взаємодії з працівниками» було описано область розробки, була детально досліджена предметна область та зроблений повний аналіз вимог. Також була розроблено моделювання та конструювання програмного забезпечення. Було детально розписано умови ринку та способи виходу на ринок та просування на ньому у розділі «Розроблення стартап проекту».

Для зручності встановлення та користування було описано процесс встановлення вебзастосунку, а також розроблена інструкція для користувача – адміністратора, користувача – співробітника та користувача – менеджера.

Розроблена система може без проблем бути доповнена та масштабована.

ПЕРЕЛІК ПОСИЛАНЬ

1) Trello [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Trello.
2) Slack [Електронний ресурс] – Режим доступу до ресурсу: https://buduysvoe.com/publications/biznes-bez-reklamy-uspih-slack
3) Microsoft Teams [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Microsoft_Teams
4) Інноваційний підхід до управління людськими ресурсами / Р.А. Кочуріна, Д.М. Магомедкерімова // Науково-методичний електронний журнал Концепт. - 2016. - № S5. - С. 36-40.
5) Стратегія управління персоналом найважливіш складова в управлінні організації / Ю.М. Короленка // Сучасн технології управління персоналом: збірник наукових праць. - Сімферополь, 2016. - С. 33-38.
6) Проблеми управління по цілям і використання КПЕ в управлінні людськими ресурсами в сучасній російській економіці / Т.В. Сабетова // Нормування і оплата праці в промисловості. - 2015. - № 7. - С. 28-33
7) Перехід управління персоналом до управління людськими ресурсами та формування міжнародної моделі HRкомпетенцій / Н.В. Климович // Вісник Університету (Державний університет управління). - 2015. - № 10. - С. 140-144.
8) Diversity management як сучасний підхід до управління людськими ресурсами: теорія і практика - 2015. - С. 182-185.
9) Основи управління персоналом / А.Я. Кібанов. - М .: Инфра-М, 2016. - 448 с.
10) Принципи та методи управління персоналом [Електронний ресурс] – Режим доступу до ресурсу: https://ukrguru.ru/biznes/101640-principi-ta-metodi-upravlinnja-personalom.html
11) MongoDB [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/MongoDB

12) Корнієнко Б.Я. Дослідження імітаційного полігону захисту критичних інформаційних ресурсів методом IRISK. Моделювання та інформаційні технології. 2018. Вип. 83. С. 34-41.
13) Корнієнко Б.Я. Побудова та тестування імітаційного полігону захисту критичних інформаційних ресурсів. Наукоємні технології. 2017. № 4 (36). С. 316 - 322.
14) Korniyenko B., Yudin A., Galata L. Risk estimation of information system.Wschodnioeuropejskie Czasopismo Naukowe. 2016. № 5. P. 35 - 40.
15) Корнієнко Б.Я., Юдін О.К., Снігур О.С.Безпека аутентифікації у web-ресурсах. Захист інформації. 2012. № 1 (54). С. 20 -25. DOI: 10.18372/2410-7840.14.2056 (ukr).
16) Корнієнко Б.Я., Максімов Ю.О., Марутовська Н.М. Прикладні програми управління інформаційними ризиками. Захист інформації. 2012. № 4 (57). С. 60 – 64. DOI: 10.18372/2410-7840.14.3493 (ukr).
17) Galata, L., Korniyenko, B., Yudin, A.: Research of the simulation polygon for the protection of critical information resources. In: CEUR Workshop Proceedings, Information Technologies and Security, Selected Papers of the XVII International Scientific and Practical Conference on Information Technologies and Security (ITS 2017), 30 Nov 2017, Kyiv, Ukraine. vol. 2067. pp. 23–31., urn:nbn:de:0074-2067-8.
18) Корнієнко Б.Я. Безпека інформаційно-комунікаційних систем та мереж. Навчальний посібник для студентів спеціальності 125 «Кібербезпека». – К.:НАУ, 2018. – 226 с.
19) Корнієнко Б.Я., Галата Л.П. Оптимізація системи захисту інформації корпоративної мережі. Математичне та комп'ютерне моделювання. Серія: Технічні науки, Випуск 19, 2019. - С. 56-62.
20) Korniyenko B., Galata L. Implementation of the information resources protection based on the CentOS operating system. Conference Proceedings of 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON -2019) July 2 – 6, 2019, Lviv, Ukraine. - pp. 1007-1011.
21) Галата Л.П., Корнієнко Б.Я., Заболотний В.В. Математична модель протидії загрозам у системі захисту критичних інформаційних ресурсів. Наукоємні технології, Том 43, № 3, 2019. – С. 300 – 306.
22) Корнієнко Б.Я. Modeling of information security system in computer network. Безпека інформаційних систем і технологій, Том №1 (1), 2019. – С.36-41.

23) Korniyenko B., Galata L., Ladieva L. Research of Information Protection System of Corporate Network Based on GNS3. Conference Proceedings of 2019 IEEE Inernational Conference on Advanced Trends in Information Theory (IEEE ATIT -2019) Dezember 18 – 20, 2019, Kyiv, Ukraine. - pp. 244-248.
24) Korniyenko B., Galata L., Ladieva L. Mathematical model of threats resistance in the critical information resources protection system. CEUR Workshop Proceedings, Selected Papers of the XIX International Scientific and Practical Conference "Information Technologies and Security" (ITS 2019) Kyiv, Ukraine, November 28, 2019. Vol-2577. P.281-291.
25) Корниенко Б.Я. Кибернетическая безопасность – операционные системы и протоколы. ISBN 978-3-330-08397-4, LAMBERT Academic Publishing, Saarbrucken, Deutschland. − 2017. – 122 с.
26) Korniyenko B.Y., Galata L.P. Design and research of mathematical model for information security system in computer network. Науковий журнал «Наукоємні технології». – 2017, № 2 (34), С. 114 - 118.
27) Корниенко Б.Я. Информационная безопасность и технологии компьютерных сетей : монография. ISBN 978-3-330-02028-3, LAMBERT Academic Publishing, Saarbrucken, Deutschland. − 2016. – 102 с.
28) Korniyenko B., Galata L., Kozuberda O. Modeling of security and risk assessment in information and communication system. Sciences of Europe. – 2016. – V. 2. – No 2 (2). – P. 61 -63.
29) Korniyenko B. The classification of information technologies and control systems. International scientific journal. – 2016. –№ 2. – P. 78 - 81.
30) Корнієнко Б.Я. Інформаційні технології оптимального управління виробництвом мінеральних добрив :монографія. – К.: Вид-во Аграр Медіа Груп, 2014. – 288 с.
31) Korniyenko B., Galata L., Ladieva L. Security Estimation of the Simulation Polygon for the Protection of Critical Information Resources / B. Korniyenko, //CEUR Workshop Proceedings, Selected Papers of the XVIII International Scientific and Practical Conference "Information Technologies and Security" (ITS 2018) Kyiv, Ukraine, November 27, 2018, Vol-2318, - P. 176-187, urn:nbn:de:0074-2318-4

ДОДАТОК А ТЕКСТИ ПРОГРАМНОГО КОДУ

Тексти програмного коду

Веб-застосунок для онлайн управління та взаємодії з працівниками

CD-ROM

(Вид носія даних)

10 арк, 1488 Кб

(Обсяг програми (документа) , арк.,)

```
const express = require("express");
const app = express();
const dotenv = require("dotenv");
const mongoose = require("mongoose");
const cors = require("cors");
const PORT = process.env.PORT || 4050;

//IMPORT ROUTES

const adminRoute = require("./routes/adminauth/adminauth");
const managerRoute = require("./routes/managerauth/managerauth");
const employeeRoute = require("./routes/employeeauth/employeeauth");
const adminDashboardRoute = require("./routes/adminauth/adminDashboard");
const managerDashboardRoute = require("./routes/managerauth/managerDashboard");
const employeeDashboardRoute = require("./routes/employeeauth/employeeDashboard");

dotenv.config();

//CONNECTION TO DATABASE

mongoose.connect(
  'mongodb://localhost',
  { useNewUrlParser: true, useUnifiedTopology: true },
  () => console.log("connected to db  ")
);

//MIDDLEWARE
app.use(express.json(), cors());

//ROUTE MIDDLEWARE

app.use("/api/admin", adminRoute);
app.use("/api/manager", managerRoute);
app.use("/api/employee", employeeRoute);
```

```
app.use("/api/admindashboard", adminDashboardRoute);
app.use("/api/managerdashboard", managerDashboardRoute);
app.use("/api/employeedashboard", employeeDashboardRoute);

app.get("/", (req, res) => {
  res.send(
    `<a href="https://github.com/shelcia/CRM-backend">This is a backend app , click to open code</a>`
  );
});
app.listen(PORT, () => console.log(`server up and running at  ${PORT}`));

// "start": "nodemon index.js"
const mongoose = require("mongoose");

const contactSchema = new mongoose.Schema({
  title: {
    type: String,
    required: true,
    min: 3,
    max: 255,
  },
  client: {
    type: String,
    required: true,
    min: 3,
    max: 255,
  },
  number: {
    type: Number,
    required: true,
    min: 3,
  },
  email: {
```

```
    type: String,
    required: true,
    max: 1024,
    min: 2,
  },
  address: {
    type: String,
    required: true,
    max: 1025,
    min: 2,
  },
  date: {
    type: Date,
    default: Date.now(),
  },
});

module.exports = mongoose.model("Contacts", contactSchema);

const mongoose = require("mongoose");

const leadSchema = new mongoose.Schema({
  title: {
    type: String,
    required: true,
    min: 3,
    max: 255,
  },
  client: {
    type: String,
    required: true,
    min: 3,
    max: 255,
  },
```

```
  number: {
    type: Number,
    required: true,
    min: 3,
  },
  status: {
    type: String,
    required: true,
    max: 255,
    min: 2,
  },
  date: {
    type: Date,
    default: Date.now(),
  },
});

module.exports = mongoose.model("Lead", leadSchema);
const mongoose = require("mongoose");

const serviceRequestSchema = new mongoose.Schema({
  title: {
    type: String,
    required: true,
    min: 3,
    max: 255,
  },
  client: {
    type: String,
    required: true,
    min: 3,
    max: 255,
  },
  manager: {
```

```
  type: String,
  required: true,
  min: 3,
  max: 255,
},
expected_revenue: {
  type: Number,
  required: true,
  min: 2,
},
probability: {
  type: Number,
  required: true,
  max: 255,
},
status: {
  type: String,
  required: true,
  max: 255,
  min: 2,
},
expected_closing: {
  type: Date,
  required: true,
},
priority: {
  type: String,
  required: true,
  max: 255,
  min: 2,
},
date: {
  type: Date,
  default: Date.now(),
```

```
  },
});

module.exports = mongoose.model("ServiceRequest", serviceRequestSchema);
const mongoose = require("mongoose");

const userSchema = new mongoose.Schema({
  fname: {
    type: String,
    required: true,
    min: 6,
    max: 255,
  },
  lname: {
    type: String,
    required: true,
    min: 6,
    max: 255,
  },
  email: {
    type: String,
    required: true,
    max: 255,
    min: 6,
  },
  password: {
    type: String,
    required: true,
    max: 1024,
    min: 6,
  },
  type: {
    type: String,
    required: true,
```

```
    max: 10,
    min: 2,
  },
  date: {
    type: Date,
    default: Date.now(),
  },
});

module.exports = mongoose.model("User", userSchema);
const router = require("express").Router();
const bcrypt = require("bcryptjs");
const jwt = require("jsonwebtoken");
const nodemailer = require("nodemailer");
const User = require("../../models/User");
const verify = require("./adminverfiy");

//VALIDATION OF USER INPUTS PREREQUISITES
const Joi = require("@hapi/joi");

const registerSchema = Joi.object({
  fname: Joi.string().min(3).required(),
  lname: Joi.string().min(3).required(),
  email: Joi.string().min(6).required().email(),
  password: Joi.string().min(6).required(),
});

const loginSchema = Joi.object({
  email: Joi.string().min(6).required().email(),
  password: Joi.string().min(6).required(),
});

//SIGNUP USER
router.post("/register", verify, async (req, res) => {
```

```
//CHECKING IF USER EMAIL ALREADY EXISTS
const emailExist = await User.findOne({ email: req.body.email });
if (emailExist) res.status(400).send("Email already exists");

//HASHING THE PASSWORD

const salt = await bcrypt.genSalt(10);
const hashedPassword = await bcrypt.hash(req.body.password, salt);

//ON PROCESS OF ADDING NEW USER

const user = new User({
  fname: req.body.fname,
  lname: req.body.lname,
  email: req.body.email,
  password: hashedPassword,
  type: "admin",
});

try {
  //VALIDATION OF USER INPUTS

  const { error } = await registerSchema.validateAsync(req.body);
  if (error) return res.status(400).send(error.details[0].message);
  else {
    //NEW USER IS ADDED

    const saveUser = await user.save();
    //   res.send({ user: user._id });
    res.send("user created");
  }
} catch (error) {
  res.status(400).send(error);
}
```

```
});

//SIGNIN USER

router.post("/login", async (req, res) => {
 //CHECKING IF USER EMAIL EXISTS

 const user = await User.findOne({ email: req.body.email });
 if (!user) return res.status(400).send("Incorrect Email- ID");

 //CHECKING IF USER PASSWORD MATCHES

 const validPassword = await bcrypt.compare(req.body.password, user.password);
 if (!validPassword)
  return res.status(400).send({ message: "Incorrect Password" });

 try {
  //VALIDATION OF USER INPUTS

  const { error } = await loginSchema.validateAsync(req.body);
  if (error)
   return res.status(400).send({ message: error.details[0].message });
  else {
   //   res.send("success");
   if (user.type === "admin") {
    const token = jwt.sign(
     { _id: user._id },
     'test'
    );
    res.status(200).header("auth-token", token).send(token);
   } else {
    res.status(200).json({ message: "Seems like you are not a admin" });
   }
  }
```

```
  } catch (error) {
    res.status(400).send(error);
  }
});

//DELETE USER

router.delete("/deleteuser", verify, async (req, res) => {
  try {
    const users = await User.deleteOne({ email: req.body.email });
    res.status(200).send("deleted suuccesfully");
  } catch (error) {
    console.log(error);
    res.status(400).send(error);
  }
});

module.exports = router;
//ADMIN DASHBOARD

//ADMIN CAN EDIT,VIEW,DELETE AND ADD

const router = require("express").Router();
const verify = require("./adminverfiy");
const ServiceRequest = require("../../models/ServiceRequest");
const Lead = require("../../models/Lead");
const Contact = require("../../models/Contact");
const User = require("../../models/User");

//VALIDATION OF USER INPUTS PREREQUISITES
const Joi = require("@hapi/joi");

const serviceRequestSchema = Joi.object({
  title: Joi.string().min(3).required(),
```

```
  client: Joi.string().min(3).required(),
  manager: Joi.string().min(3).required(),
  expected_revenue: Joi.number().min(2).required(),
  probability: Joi.number().required(),
  status: Joi.string().min(2).required(),
  expected_closing: Joi.date().required(),
  priority: Joi.string().min(2).required(),
});

//SERVICE REQUEST API'S

//POST
router.post("/servicerequest", verify, async (req, res) => {
  const ticket = new ServiceRequest({
    title: req.body.title,
    client: req.body.client,
    manager: req.body.manager,
    expected_revenue: req.body.expected_revenue,
    probability: req.body.probability,
    status: req.body.status,
    expected_closing: req.body.expected_closing,
    priority: req.body.priority,
  });

  try {
    //VALIDATION OF USER INPUTS

    const { error } = await serviceRequestSchema.validateAsync(req.body);
    if (error) return res.status(400).send(error.details[0].message);
    else {
      //NEW SERVICE REQUEST  IS ADDED

      const saveTicket = await ticket.save();
      res.send("service request created");
```

```
    }
  } catch (error) {
    res.status(400).send(error);
  }
});

//GET

router.get("/servicerequest", verify, async (req, res) => {
  try {
    const tickets = await ServiceRequest.find().exec();
    res.status(200).send(tickets);
  } catch (error) {
    console.log(error);
    res.status(400).send(error);
  }
});

//DELETE

router.delete("/servicerequest", verify, async (req, res) => {
  try {
    const tickets = await ServiceRequest.deleteOne({ _id: req.body._id });
    res.status(200).send("deleted suuccesfully");
  } catch (error) {
    console.log(error);
    res.status(400).send(error);
  }
});

//PUT

router.put("/servicerequest/:id", async (req, res) => {
  try {
```

```
    const tickets = await ServiceRequest.findById(req.params.id).exec();
    tickets.set(req.body);
    const result = await tickets.save();
    res.send(result);
  } catch (error) {
    res.status(500).send(error);
  }
});

//VALIDATION OF USER INPUTS PREREQUISITES

const LeadSchema = Joi.object({
  title: Joi.string().min(3).required(),
  client: Joi.string().min(3).required(),
  number: Joi.number().min(3).required(),
  status: Joi.string().min(2).required(),
});

//LEAD API'S

//POST

router.post("/lead", verify, async (req, res) => {
  const lead = new Lead({
    title: req.body.title,
    client: req.body.client,
    number: req.body.number,
    status: req.body.status,
  });

  try {
    //VALIDATION OF USER INPUTS

    const { error } = await LeadSchema.validateAsync(req.body);
```

```
    if (error) return res.status(400).send(error.details[0].message);
    else {
      //NEW LEAD IS ADDED

      const leads = await lead.save();
      res.send("Lead created");
    }
  } catch (error) {
    res.status(400).send(error);
  }
});

//GET

router.get("/lead", verify, async (req, res) => {
  try {
    const leads = await Lead.find().exec();
    res.status(200).send(leads);
  } catch (error) {
    console.log(error);
  }
});

//DELETE

router.delete("/lead", verify, async (req, res) => {
  try {
    const leads = await Lead.deleteOne({ _id: req.body._id });
    res.status(200).send("deleted suuccesfully");
  } catch (error) {
    console.log(error);
    res.status(400).send(error);
  }
});
```

```
//PUT

router.put("/lead/:id", async (req, res) => {
  try {
    const leads = await Lead.findById(req.params.id).exec();
    leads.set(req.body);
    const result = await leads.save();
    res.send(result);
  } catch (error) {
    res.status(500).send(error);
  }
});

//VALIDATION OF USER INPUTS PREREQUISITES

const ContactSchema = Joi.object({
  title: Joi.string().min(3).required(),
  client: Joi.string().min(3).required(),
  email: Joi.string().min(2).required().email(),
  number: Joi.number().min(3).required(),
  address: Joi.string().min(2).required(),
});

//CONTACT API'S

//POST

router.post("/contact", verify, async (req, res) => {
  const contact = new Contact({
    title: req.body.title,
    client: req.body.client,
    email: req.body.email,
    number: req.body.number,
```

```
  address: req.body.address,
 });

 try {
  //VALIDATION OF USER INPUTS

  const { error } = await ContactSchema.validateAsync(req.body);
  if (error) return res.status(400).send(error.details[0].message);
  else {
   //NEW CONTACT IS ADDED

   const contacts = await contact.save();
   res.send("Contact created");
  }
 } catch (error) {
  res.status(400).send(error);
 }
});
```